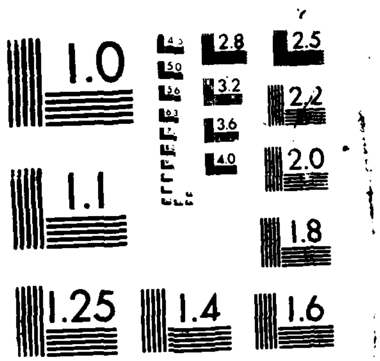
AD-A186 525 A RAYTRACE METHOD FOR A LATERALLY HETEROGENEOUS ENVIRONMENT(U) NAVAL OCEAN SYSTEMS CENTER SAN DIEGO CA W L PATTERSON JUL 87 NOSC/TD-1180 1/1

UNCLASSIFIED F/G 20/14 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

NAVAL OCEAN SYSTEMS CENTER San Diego, California 92152-5000

NOSC TR 1180

Technical Report 1180
July 1987

A Raytrace Method for a Laterally Heterogeneous Environment

W. L. Patterson

NAVAL OCEAN SYSTEMS CENTER
San Diego, California 92152-5000

E. G. SCHWEIZER, CAPT, USN
Commander

R. M. HILLYER
Technical Director

ADMINISTRATIVE INFORMATION

The work described in this report was completed for the Office of Naval Technology (ONT) by Code 543 of the Naval Ocean Systems Center.

Released by
H.V. Hitney, Head
Tropospheric Branch

Under authority of
J.H. Richter, Head
Ocean and Atmospheric
Sciences Division

JJ

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

ADA186525

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NOSC TD 1180	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Ocean Systems Center 6b OFFICE SYMBOL (if applicable)	7a NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State and ZIP Code) San Diego, CA 92152-5000	7b ADDRESS (City, State and ZIP Code)
8a NAME OF FUNDING SPONSORING ORGANIZATION Office of Naval Technology 8b OFFICE SYMBOL (if applicable) ONT	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State and ZIP Code) Arlington, VA 22217	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	AGENCY ACCESSION NO
62759N	SXB3		DN888 715

11 TITLE (include Security Classification)
A Raytrace Method for a Laterally Heterogeneous Environment

12 PERSONAL AUTHOR(S)
W.L. Patterson

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Final	FROM Oct 86 TO Apr 87	July 1987	64

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	raytrace technique isotropic homogeneous medium

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A raytrace technique has been developed to show the wave front path as it propagates through this laterally heterogeneous medium where the index of refraction is allowed to vary both vertically and horizontally.

20 DISTRIBUTION AVAILABILITY OF ABSTRACT [ ] UNCLASSIFIED/UNLIMITED [X] SAME AS RPT [ ] DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL W.L. Patterson	22b TELEPHONE (Include Area Code) (619)225-7247	22c OFFICE SYMBOL Code 543

DD FORM 1473, 84 JAN

83 APR EDITION MAY BE USED UNTIL EXHAUSTED
ALL OTHER EDITIONS ARE OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

EXECUTIVE SUMMARY

OBJECTIVE

The objective is to provide environmental support to system designers and field commanders to accurately forecast pertinent conditions and to make real time assessment/ tactical conditions.

RESULTS

A raytrace technique has been developed to show the wave front path as it propagates through the laterally heterogeneous medium where the index of refraction is allowed to vary both vertically and horizontally.

RECOMMENDATION

This effort is specially geared toward future Navy systems that are presently in the planning stage or under development.

CONTENTS

Page

INTRODUCTION 1

BACKGROUND 1

RAY THEORY 3

LINEAR MODEL OF A HETEROGENEOUS ATMOSPHERE 5

OPERATOR PROGRAM INTERACTIONS 7

- Data Requirements 7
- Environmental Data Considerations 7
- Layer Boundary Determination 9

RAYTRACE TECHNIQUE 10

RAYTRACE EXAMPLES 12

REFERENCES 20

APPENDIX A .... *Ai*

- Program: Laterally Heterogeneous Atmosphere Raytrace A1
- Subroutine: envidata A7
- Subroutine: sysdata A10
- Subroutine: drawprof A12
- Subroutine: drawlayers A16
- Subroutine: drawrange A17
- Subroutine: refcond A23
- Subroutine: newdata A24
- Subroutine: raytrace A28
- Subroutine: gradient A35

ILLUSTRATIONS

Figure

1. Relative bending for four types of refractive conditions 2
2. Refraction at a boundary 3
3. Refraction through an infinitesimally changing n. 3
4. Illustration of Snell's law for a curved earth 4
5. Space cross section in B-units, 12 March 1948, San Diego, California seaward to Guadalupe Islands. $\left(B = \left[(n-1) + \frac{z}{4a}\right] 10^6\right)$ 6
6. Space cross section in N units, 1500Z, 19 February 1952. Solid vertical line represents position of mid-latitude wave cyclone cold front 6

ILLUSTRATIONS (Continued)

Figure		Page
7.	M-unit versus height profiles, 16 September, 1985, for stations of Deerfield, Salem and Hanscom AFB, NH	6
8.	An example heterogeneous atmosphere defined by 5 M-unit versus height profiles	9
9.	Example layer boundaries	10
10.	Geometry for a ray crossing an environmental layer's boundary	11
11.	Ray family for laterally homogeneous trapping layer	12
12.	Ray family for a rising trapping layer	13
13.	Ray family for a falling trapping layer	13
14.	Family of ray paths calculated for a slope of 1.89×10^{-3}, a gradient in the interface of -0.1 N-units per ft, and a gradient above and below the interface of -0.01 N-units per ft	15
15.	Family of ray paths calculated for a 500 ft thick interface with a slope of 1.89×10^{-3}, a gradient in the interface of -0.1 N-units per ft, and a gradient above and below the interface of -0.01 N-units per ft	15
16.	Ray family for environment of figure 5. Transmitter height of 500 ft	16
17.	Ray family for environment of figure 5. Transmitter height of 1000 ft	16
18.	Ray family within a surface-based trapping layer of decreasing thickness	19
19.	Ray family within an elevated trapping layer of variable thickness	19

TABLES

Table		
1.	The relationship between *N*, *M*, and refractive conditions	1
2.	Environmental and transmitter data used in figure 11	12
3.	Environmental and transmitter data used in figure 12	14
4.	Environmental and transmitter data used in figure 13	14
5.	Environmental and transmitter data used in figures 16 and 17	17
6.	Environmental and transmitter data used in figures 18 and 19	18

INTRODUCTION

Radio waves travel in straight lines through any isotropic, homogeneous medium with a velocity which depends upon the speed of light in a vacuum and the refractive index of the medium. If the index of refraction changes, the velocity and possibly the direction of travel will change.

Over major land and ocean surfaces, conditions of temperature and moisture are sufficiently homogeneous to cause the formation of air masses which provide a laterally homogeneous medium for electromagnetic wave propagation. Since air mass conditions are the norm for most of the troposphere, many raytrace techniques have been developed to illustrate a wave front as it propagates through this laterally homogeneous medium.

At air mass boundaries associated with wave cyclones, Foehn circulations, land ocean interfaces, etc., the conditions of lateral homeogeneity often do not exist. A raytrace technique has been developed to show the wave front path as it propagates through this laterally heterogeneous medium where the index of refraction is allowed to vary both vertically and horizontally. Appendix A lists the program code (Microsoft QuickBASIC version 2.01) for the raytrace technique as employed upon an IBM PC, XT, AT or compatible using the MS-DOS version 3.2 operating system.

BACKGROUND

The refractive index n of a parcel of air is defined as the ratio of the velocity of propagation of an electromagnetic (EM) wave in a vacuum to that in the air. Since the refractive index of the atmosphere is slightly greater than unity, EM waves travel slightly slower in air than in a vacuum. Close to the earth's surface, the numeric value of the refractive index is usually between 1.00025 and 1.0004. For convenience, the refractivity N is defined by Bean and Dutton (1968) as

$$N = (n - 1)\,10^6 \qquad (1)$$

such that surface values of refractivity vary between 250 and 400. Refractivity may be expressed as a function of atmospheric pressure, temperature, and humidity by the relationship

$$N = \frac{77.6\ P}{T} + \frac{3.73\ 10^5 e}{T^2} \qquad (2)$$

where

P = atmospheric pressure (millibars)
T = atmospheric temperature (Kelvin)
e = atmospheric water vapor pressure (millibars).

A ray path under well-mixed (standard) atmospheric conditions will bend downward at a rate less than the curvature of the earth. Certain atmospheric conditions could also lead to the ray path bending downward at a rate exceeding the curvature of the earth (trapping), downward more than standard but not sufficient for trapping (super-refraction), or upward (sub-refraction). Figure 1 illustrates these refractive conditions.

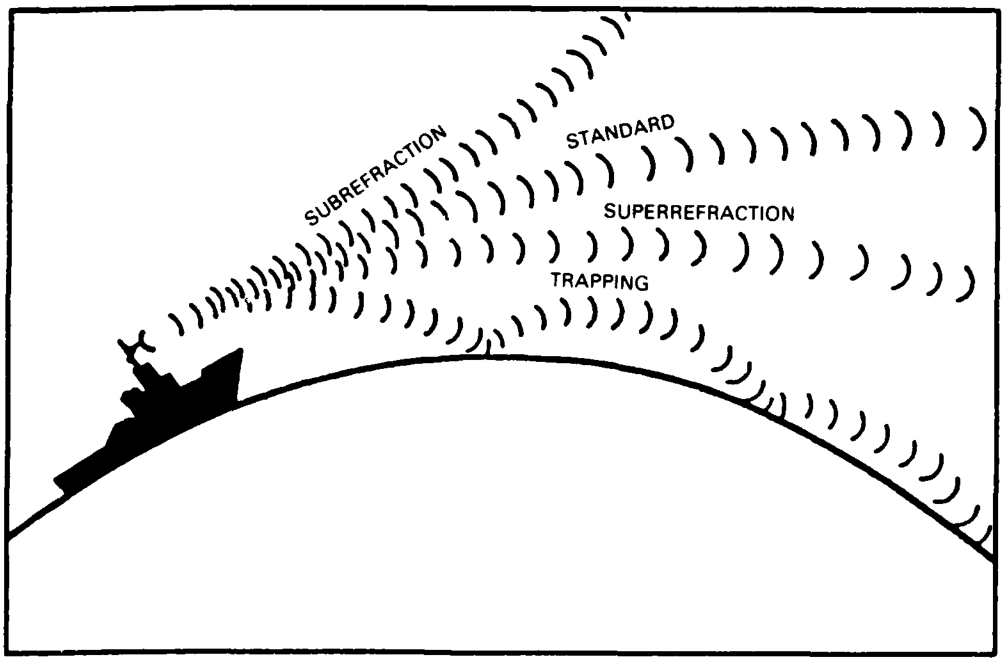


Figure 1. Relative bending for four types of refractive conditions.

As a further convenience in computing and graphically displaying these types of refractive conditions, a modified refractivity M may be defined as

$$M = \left(n - 1 + \frac{z}{a}\right)10^6 \tag{3}$$

where

a = mean earth's radius and

z = specified height above the earth's surface.

The advantage of the modified refractivity is that trapping environments are indicated with a negative M-unit gradient with increasing height. Table 1 lists the relations between N, M, and the refractive conditions.

Table 1. The relationship between N, M, and refractive conditions.

Types of refraction	N-unit gradient (N/km)	M-unit gradient (M/km)
Trapping	≤ −157	≤ 0
Super-refraction	−157 to −79	0 to 79
Standard	−79 to 0	79 to 157
Sub-refraction	> 0	> 157

RAY THEORY

The propagation of EM waves within the troposphere may be simply explained by ray theory as developed by Reed and Russell (1966). As stat‹ ‿ above, the refractive conditions lead to a change in the propagating ray's direction. Figure 2 illustrates the direction change and is defined by Snell's law:

$$n_1 \cos(\alpha_1) = n_2 \cos(\alpha_2) \tag{4}$$

where the boundary is a plane surface.

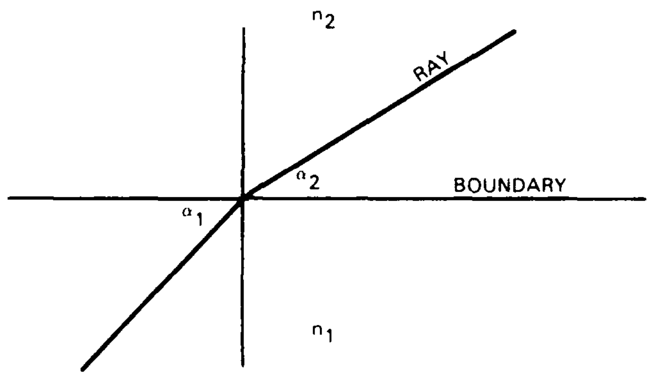


Figure 2. Refraction at a boundary.

In the atmosphere, the refractive index changes continuously or may be thought to change linearly when the atmosphere is divided into an infinite number of parallel boundaries and the refractive index changes by infinitesimal amounts. As illustrated in figure 3, Snell's law may now be written as

$$n_1 \cos(\alpha_1) = n_0 \cos(\alpha_0) \tag{5}$$

where n_1 and α_1 are functions of height and n_0 and α_0 are fixed values at a given reference.

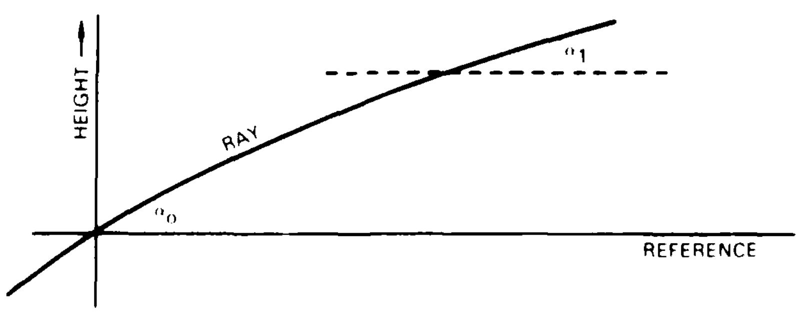


Figure 3. Refraction through an infinitesimally changing n.

Since the earth's surface is spherical and not planar, Snell's law may be rewritten as

$$n_1 r_1 \cos(\alpha_1) = n_0 r_0 \cos(\alpha_0) \tag{6}$$

where

r_1 and r_0 are the distances from the earth's center to the boundaries and α_1 and α_0 are the angles between the ray and the planes normal to the radius vector at the points where the ray crosses the boundaries as shown by figure 4.

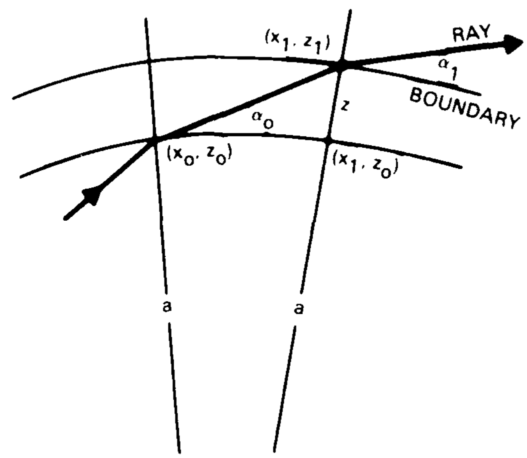


Figure 4. Illustration of Snell's law for a curved earth.

If z is the height of an antenna above the earth's surface and a is the radius of the earth, then

$$r_1 = z + a \quad \text{and} \quad r_0 = a \tag{7}$$

and equation 6 may be written as

$$n_1 \left(1 + \frac{z}{a}\right) \cos(\alpha_1) = n_0 \cos(\alpha_0) . \tag{8}$$

With the assumption that n is close to unity and $\frac{z}{a}$ is a vary small quantity compared to unity

$$n \left(1 + \frac{z}{a}\right) = n + \frac{z}{a} \tag{9}$$

and for small values of α

$$\cos(\alpha) = 1 - \frac{\alpha^2}{2} \tag{10}$$

Snell's law may be written as

$$\frac{1}{2}\left(\alpha_1^2 - \alpha_2^2\right) = n_1 - n_0 + \frac{z}{a} \, . \tag{11}$$

Substituting the modified refractivity for the refractive index, solving for the ending height and angle of a ray when given the beginning position, and the ray's initial launch angle yields the traditional raytrace equations employed by well-established raytracing theory. These equation are

$$\alpha_1 = \alpha_0 + \frac{\Delta M}{\Delta z}\,(x_1 - x_0)\, 10^{-6} \tag{12}$$

and

$$z_1 = z_0 + \frac{\alpha_1^2 - \alpha_0^2}{\frac{\Delta M}{\Delta z}\, 10^{-3}} \, . \tag{13}$$

where

α_0 = ray angle at start of calculation (radians)

α_1 = ray angle at end of calculation (radians)

z_0 = height at start of calculation (meters)

z_1 = height at end of calculation (meters)

x_0 = range at start of calculation (kilometers)

x_1 = range at end of calculation (kilometers)

$\frac{\Delta M}{\Delta z}$ = change in M-units with respect to a change in height (M units per meter).

LINEAR MODEL OF A HETEROGENEOUS ATMOSPHERE

To model an atmosphere which varies both vertically and horizontally, it is necessary to make assumptions concerning the behavior of the refractive index. At air mass boundaries caused by large scale subsidence, such as within the trade wind inversion as illustrated by Hitney, et al. (1985) in figure 5, it is natural to assume a horizontally homogeneous continuation of the boundary. With vertical air mass boundaries, such as those associated with mid-latitude wave cyclones as shown by Bean and Dutton (1968) in figure 6, or those associated with radiational cooling as shown by Morrissey (1985) in figure 7, the horizontal variation of refractivity may be more complicated.

In this raytrace technique, it is assumed the atmosphere can be divided into layers with piece-wise linear variations in the layer height and piece-wise linear refractivity within each layer. In areas where a duct degenerates to a linear profile or a duct splits to form several detached ducts, layer boundaries are determined by the meteorological conditions.

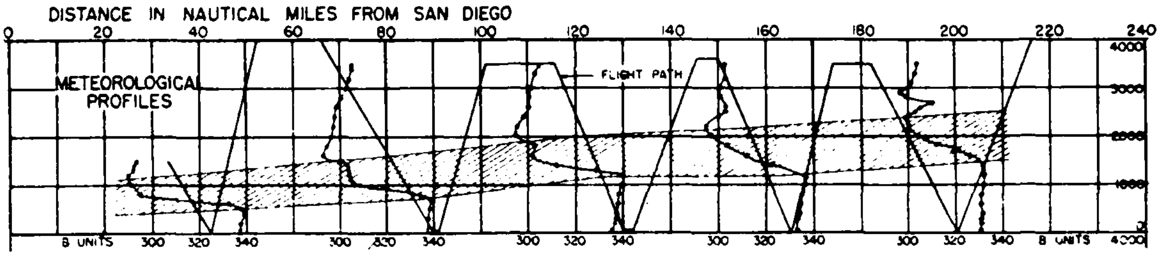


Figure 5. Space cross section in B-units, 12 March 1948, San Diego, CA seaward to Guadalupe Islands. $\left(B = \left[(n - 1) + \frac{z}{4a} \right] 10^6. \right)$

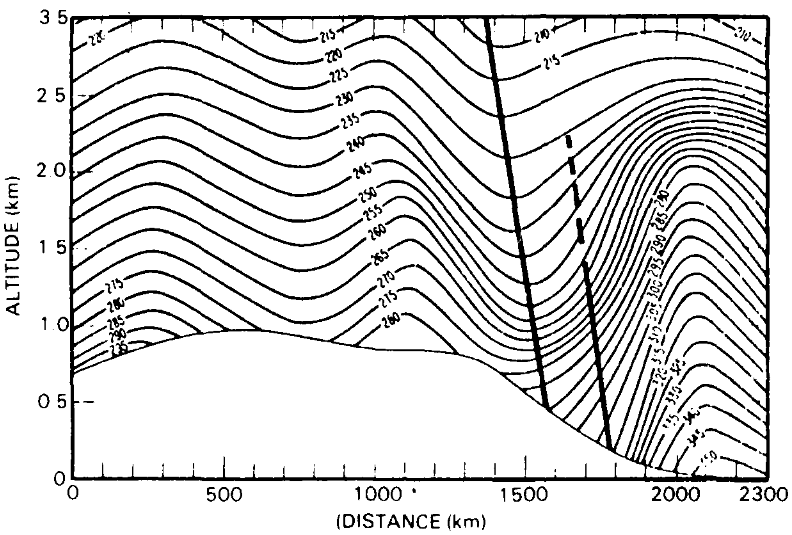


Figure 6. Space cross section in N units, 1500Z, 19 February 1952. Solid vertical line represents position of mid-latitude wave cyclone cold front.

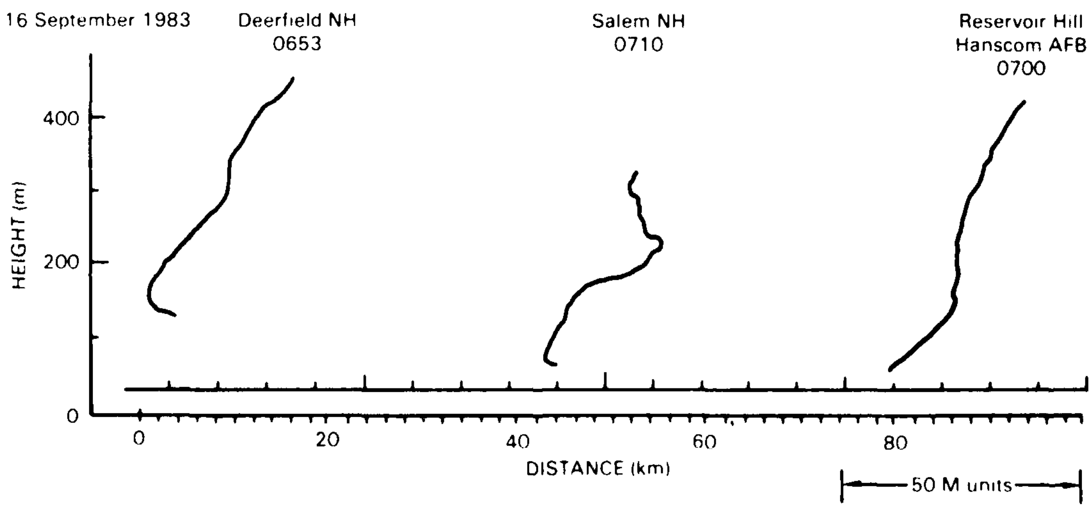


Figure 7. M-unit versus height profiles, 16 September, 1985, for stations of Deerfield, Salem and Hanscom AFB, NH.

Knowing the vertical structure of refractivity and the horizontal distance between two refractivity profiles "a" and "b," the layer boundary height z, and the refractivity gradient $\frac{\Delta M}{\Delta z}$ is given by the relationship

$$z_x = \frac{(z_b - z_a)}{(x_b - x_a)} x + z_a \tag{14}$$

and

$$\frac{\Delta M}{\Delta z}_{(x,z)} = \frac{\left(\frac{\Delta M}{\Delta z}_b (z) - \frac{\Delta M}{\Delta z}_a (z)\right)}{(x_b - x_a)} x + \frac{\Delta M}{\Delta z}_a (z) \ . \tag{15}$$

OPERATOR / PROGRAM INTERACTIONS

DATA REQUIREMENTS

The operator is allowed to interact with the program in two ways. First, the operator may enter data from the keyboard or request the data be read from a disk file previously created. The input data are

a. environmental data consisting of triplets of pressure, temperature, and relative humidity, pairs of N-units and height, or pairs of M-units and height.

b. transmitter height

c. radiating antenna beamwidth

d. radiating antenna elevation angle

e. the number of rays to draw

Second, the operator may examine the refractivity profiles to manually determine the layer boundaries.

ENVIRONMENTAL DATA CONSIDERATIONS

For this program, the maximum number of environmental profiles which may be entered is five. This number was chosen for graphics convenience only. This raytrace technique may be applied to any number of profiles needed to fully describe the laterally heterogeneous atmosphere.

Prior to the determination of layer boundaries, the environmental data profiles must be modified to insure each profile starts at a height of zero and each profile terminates at the same elevation. This procedure is accomplished by the following steps:

a. If the environmental data is entered as triplets of pressure, temperature, and relative humidity, convert the data to M-units (equations 2 and 3), and height using the relationship derived by Berry (1945):

$$P = p_o \left(1 - \frac{\beta z}{T_o}\right)^{\frac{g}{R\beta}} \tag{16}$$

where

P = atmospheric pressure at a height of z

P_o = atmospheric pressure at a height of zero

β = atmospheric lapse rate

T_o = temperature at a height of zero

g = gravity

R = moist atmospheric gas constant

z = height.

b. If the environmental data is entered as pairs of N-units and height convert the data to M-units (equation 3).

c. If necessary, the lowest height within each profile is set to zero and a surface M-unit value is extrapolated using the standard atmospheric M-unit gradient of 118 M-units per kilometer as specified by Bean and Dutton (1968).

d. If necessary, extend the profile height to the height of the highest profile (z_{max}) *and determine a corresponding M-unit value from the exponential atmosphere model of* Bean and Dutton (1968):

$$M_{z_{max}} = 157 + M_{sfc}\, e^{\left[ln\left(\frac{M_{sfc}}{M_{km} - 157}\right) z_{max}\right]} \tag{17}$$

where

M_{sfc} = M-unit value at a height of zero

M_{km} = M-unit value at a height of 1 kilometer.

e. An M-unit gradient is determined at each profile height using the relationship

$$\frac{\Delta M}{\Delta z_i} = \frac{(M_{i+1} - M_i)}{(z_{i+1} - z_i)} \tag{18}$$

where the M-unit gradient at the profile top (z_{max}) is given by differentiating equation 17 with respect to height

$$\frac{\Delta M}{\Delta z}(z_{max}) = 157 \quad M_{sfc} \; ln \left(\frac{M_{sfc}}{M_{km} \quad 157} \right) e^{-\, ln \left(\frac{M_{sfc}}{M_{km} \quad 157} \right) z_{max}} \; . \qquad (19)$$

LAYER BOUNDARY DETERMINATION

The modified refractivity versus height profiles are computed and displayed on the terminal screen as illustrated in figure 8. Through screen prompts and the use of a moveable cursor, the operator is allowed to specify the layer boundaries. Each profile must contain an equal number of layers. Figure 9 illustrates the operator specified boundaries for the complicated profile set of figure 8. Since the layer boundaries play a major role in this raytrace method, the operator must insure the boundaries represent the most reasonable and meteorologically correct assessment. As demonstrated in figure 8, the elevated and surface-based trapping layers extend throughout all the profiles allowing for a simple assessment. The mid-level trapping layer however, is absent from profiles 1 and 2. Since the trapping layer was decreasing in thickness from profiles 5 to 3, the operator chose to eliminate the layer by allowing the M-unit gradient to change from standard to trapping between profiles 2 and 3. Several methods of automating this boundary determination process, including the use of potential temperature, are the subject of further investigation.

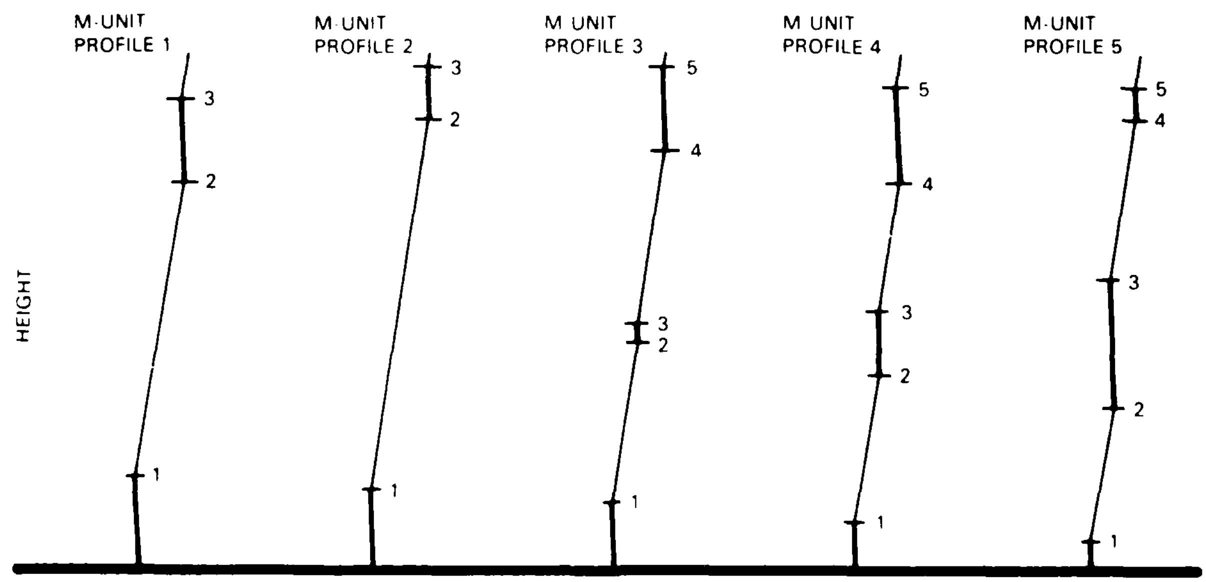


Figure 8. An example heterogeneous atmosphere defined by 5 M-unit versus height profiles.

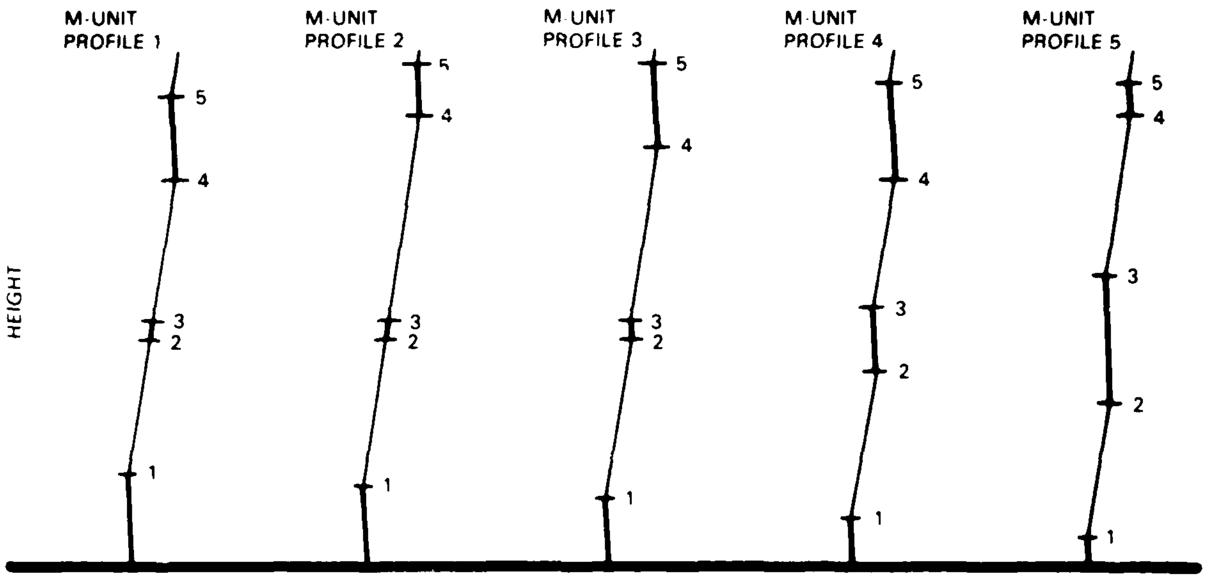


Figure 9. Example layer boundaries.

RAYTRACE TECHNIQUE

The following steps are taken in the execution of the raytrace:

a. Establish a beginning range (x_0) of zero and a beginning height (z_0) equal to the transmitter height. The first ray's beginning launch angle (α_0) is set to half of the beamwidth below the antenna elevation angle.

b. Compute a M-unit gradient at this beginning point by using equation 15.

c. Using equations 12 and 13, compute an ending height (z_1) and angle (α_1), by using as an ending range (x_1), the starting range plus an increment equal to 1/25th of the total range between the profiles being considered. The increment of range was chosen to produce a smooth curve appearance of the ray upon the graphical program output. In subsequent range steps, the ending range must be compared to the total range between profiles. If the ending range exceeds the range between profiles, the ending range is set to this total range with a new ending height and angle being computed.

d. The ending angle is examined. If it shows a sign reversal from the starting angle, the ray has passed through a maximum or minimum point. Should this be the case, the ending angle is set to zero and a range and height of this maximum or minimum is computed using equations 12 and 13. As started in (c) above, the ending range must be compared to the total range between the profiles being considered and any necessary adjustments made.

e. At the ending range, the layer's upper and lower boundary heights are computed and compared to the ending height. If the ending height is outside the layer, the range (x_i) and the height (z_i) of the ray boundary interception are computed with the relationship

$$x_i = x_0 + \Delta x \quad \text{and} \quad z_i = z_0 + \Delta z \tag{20}$$

where

$$\Delta x = -b \pm (b^2 - 4ac)^{0.5} \tag{21}$$

$$\Delta z = \{\phi(x_0 + \Delta x) + y\} - z_0 \tag{22}$$

$$a = \frac{\Delta M}{\Delta z}\left(\frac{1}{2 \cdot 10^{-3}}\right) \tag{23}$$

$$b = \left\{\frac{2\alpha_0}{2 \cdot 10^{-3}} - \phi\right\} \tag{24}$$

$$c = z_0 - \phi x_0 - y \tag{25}$$

ϕ = slope of the boundary

y = height of the boundary at the profile range.

Figure 10 illustrates the geometry and definition of terms at the interception point. Since the solution for Δx consists of two possibilities, the appropriate choice is the Δx which is positive and less than x_1.

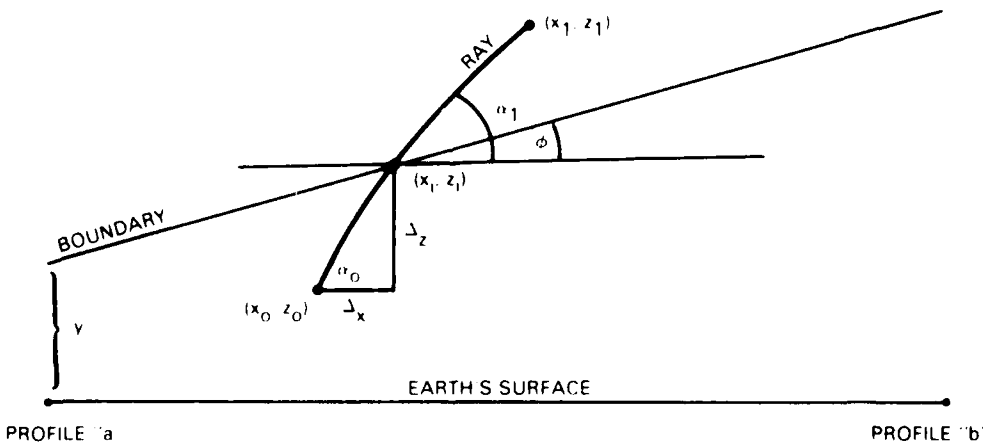


Figure 10. Geometry for a ray crossing an environmental layer's boundary.

f. The ending height is examined. If it is zero, the ray has reached the ground. In this case, the sign of the ending angle is reversed to indicate a surface reflection of the ray.

g. The beginning angle, height, and range are reinitialized with the ending angle, height, and range and steps b through g are repeated. This process continues until either the

ray has reached the maximum range between the first and last profile or has reached the maximum height for the environmental profiles. At that time, step a is repeated for the next ray, increasing the initial ray's launch angle with an increment equal to the total beamwidth divided by the number of rays to be drawn.

RAYTRACE EXAMPLES

Figure 11 illustrates the raytrace technique under a homogeneous environment with a single elevated trapping layer between 1000 and 1800 feet. Table 2 lists the transmitter and environmental data. This example is equivalent to one derived from any standard raytrace technique.

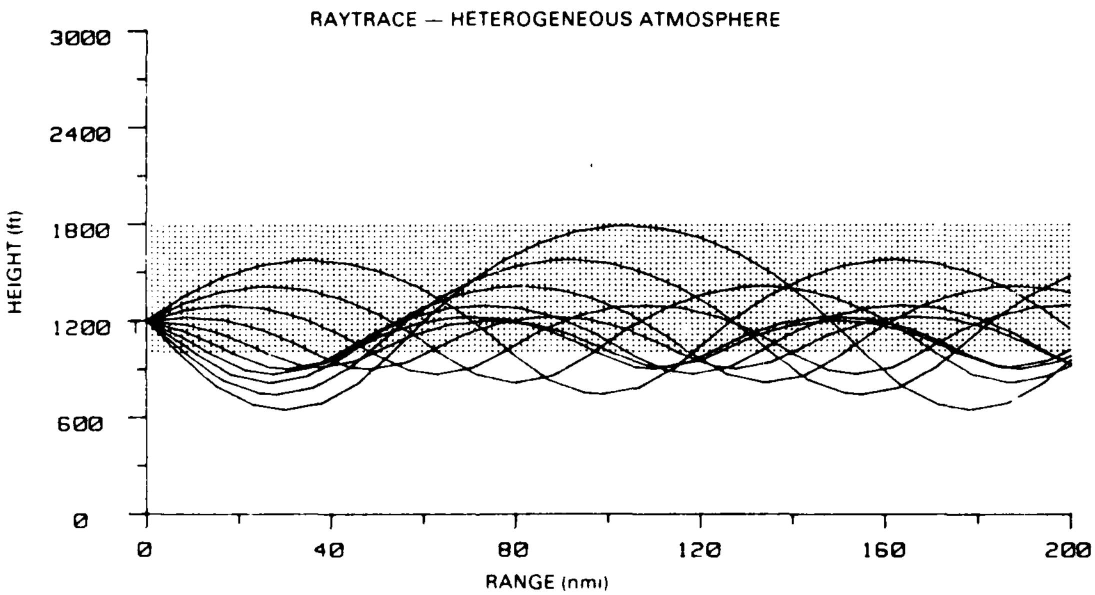


Figure 11. Ray family for laterally homogeneous trapping layer.

Table 2. Environmental and transmitter data used in figure 11.

Height (ft)	M-units
0.0	335.0
1000.0	371.0
1800.0	353.0
3000.0	412.0
Transmitter height = 1200 ft Antenna beamwidth = 0.5° Antenna elevation angle = 0°	

Figures 12 and 13 illustrate the raytrace for a transmitter located within a single trapping layer, where the layer's elevation is allowed to rise and fall, respectively, with range. The transmitter and environmental data are listed in tables 3 and 4. For comparision purposes, these data are identical to that used by Guinard, et al. (1965), in their work with rising and falling ducts and as illustrated in figures 14 and 15. There exists good agreement between the traces obtained from this technique and that described by Guinard. The major advantage of this raytrace technique over that described by Guinard is there is no need for a secondary raytrace technique near a "caustic." A caustic is defined as the point at which the rays undergo a focusing and the assumptions for ray optics theory are not valid.

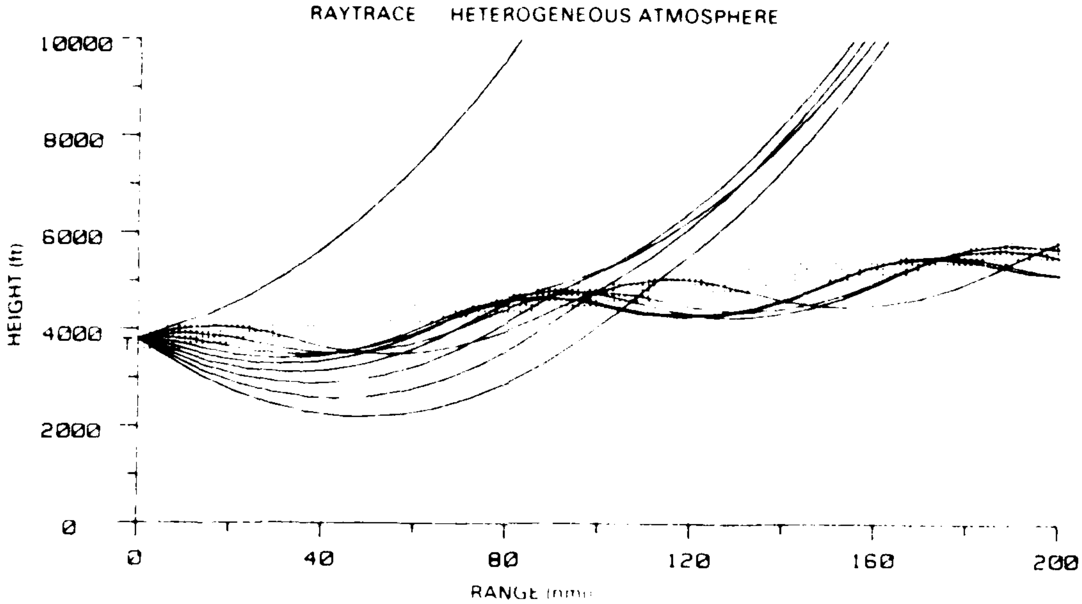


Figure 12 Ray family for a rising trapping layer

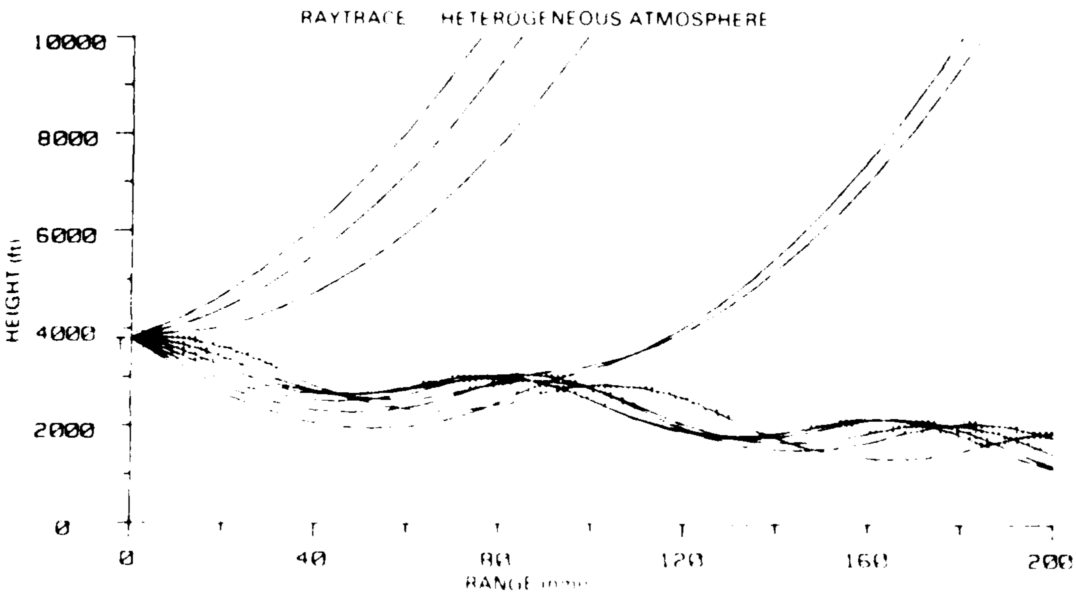


Figure 13 Ray family for a falling trapping layer

Table 3. Environmental and transmitter data used in figure 12.

Profile 1 Height (ft)	Profile 1 M-units	Profile 2 Height (ft)	Profile 2 M-units
0.0	350.0	0.0	350.0
3500.0	483.0	5500.0	559.0
4000.0	457.0	6000.0	553.0
10000.0	685.0	10000.0	685.0

Transmitter height = 3800 ft
Antenna beamwidth = 1.0°
Antenna elevation angle = 0°

Table 4. Environmental and transmitter data used in figure 13.

Profile 1 Height (ft)	Profile 1 M-units	Profile 2 Height (ft)	Profile 2 M-units
0.0	350.0	0.0	350.0
3500.0	483.0	1500.0	407.0
4000.0	457.0	2000.0	381.0
10000.0	685.0	10000.0	685.0

Transmitter height = 3800 ft
Antenna beamwidth = 1.0°
Antenna elevation angle = 0°

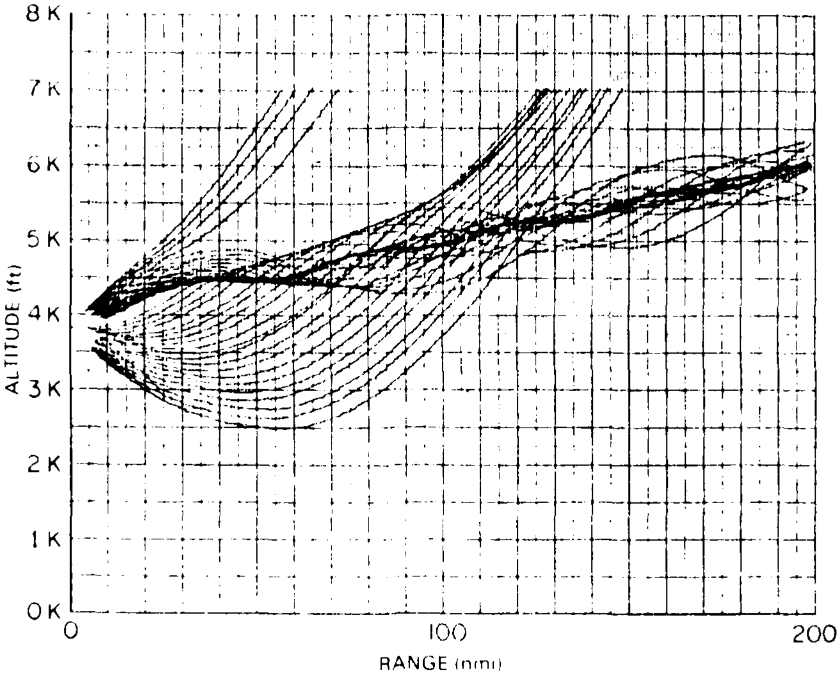


Figure 14 Family of ray paths calculated for a slope of 1.89 x 10^{-3}, a gradient in the interface of -0.1 N-units per ft, and a gradient above and below the interface of 0.01 N-units per ft.

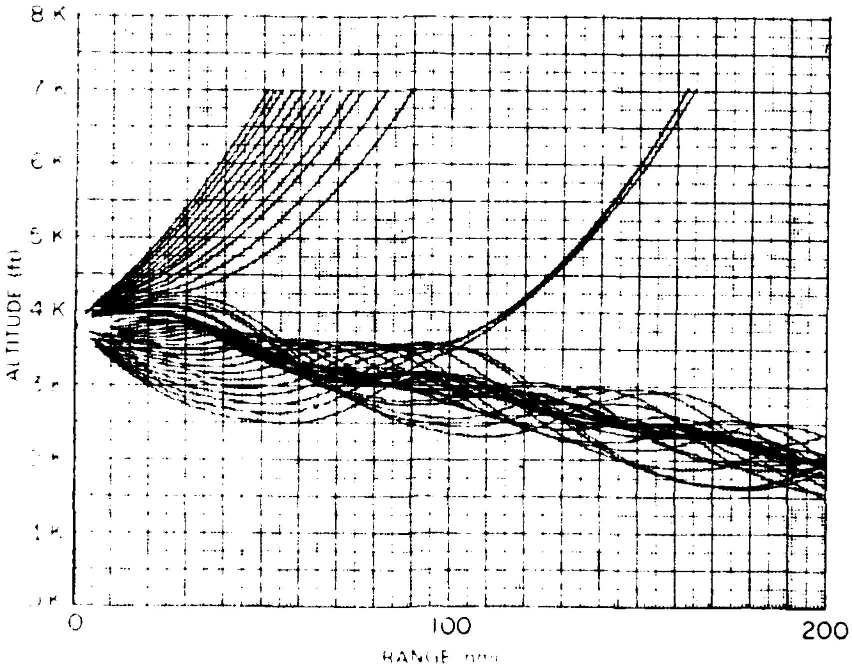


Figure 15 Family of ray paths calculated for a 500 ft thick interface with a slope of 1.89 x 10^{-3} a gradient in the interface of 0.1 N-units per ft and a gradient above and below the interface of 0.01 N-units per ft.

Figures 16 and 17 illustrate the raytrace using actual measured atmospheric conditions as shown in figure 5. For simplicity of demonstration, the measured M-unit versus height data were reduced to a set of simple trilinear profiles. Table 5 lists these data. It can be seen that when the transmitter is located at a boundary (500 feet in figure 16), the raytrace technique is able to function without special considerations when the ray launch angle and boundary slope are parallel to the earth's surface and equal to zero.

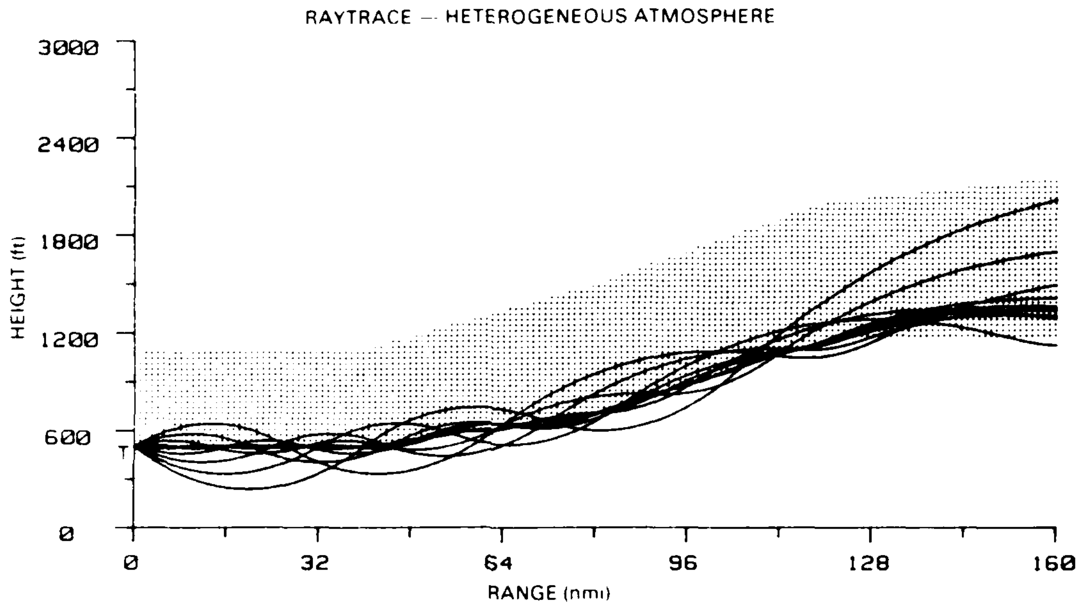


Figure 16 Ray family for environment of figure 5. Transmitter height of 500 ft.

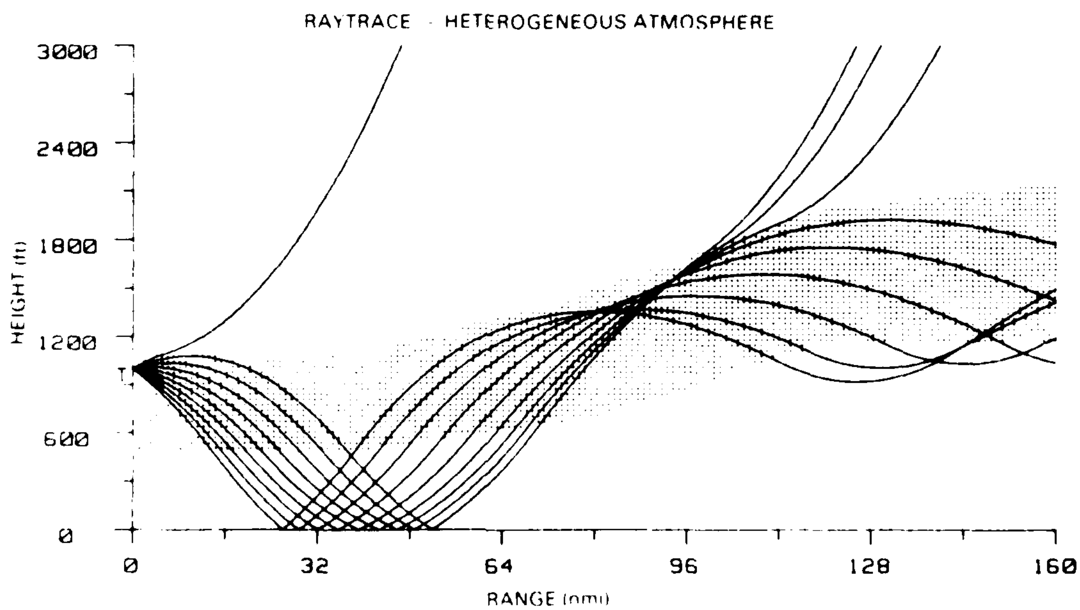


Figure 17 Ray family for environment of figure 5 Transmitter height of 1000 ft

Table 5. Environmental and transmitter data used in figures 16 and 17.

Profile 1 Height (ft)	Profile 1 M-units	Profile 2 Height (ft)	Profile 2 M-units
0.0	335.0	0.0	335.0
500.0	353.0	500.0	353.0
1100.0	327.0	1100.0	327.0
3000.0	418.0	3000.0	418.0

Profile 3 Height (ft)	Profile 3 M-units	Profile 4 Height (ft)	Profile 4 M-units
0.0	335.0	0.0	335.0
700.0	360.0	1150.0	376.0
1500.0	346.0	2000.0	364.0
3000.0	418.0	3000.0	412.0

Profile 5 Height (ft)	Profile 5 M-units
0.0	335.0
1150.0	376.0
2150.0	370.0
3000.0	411.0

Transmitter height = 500 and 1000 ft
Antenna beamwidth = 0.5°
Antenna elevation angle = 0°
Distance between profiles = 40 nmi

A limitation to the practical application of any heterogeneous atmosphere raytracing technique lays in obtaining sufficient environmental data to adequately describe the refractive conditions. For this reason, a simulated environment, as shown by figure 8 and as listed in table 6, is used to demonstrate the ability of the raytrace technique to handle a complicated environment. Figure 18 represents a transmitter located within a surface-based trapping layer which decrease in thickness with range. It may be seen that an increasing penetration angle compared to a decreasing boundary height gives rise to energy "leaking" from the trapping layer. In figure 19, the distance between the fourth and fifth profiles has been increased to more vividly demonstrate that a ray lanched within an elevated trapping layer is capable of leaving the layer and being trapped within another layer.

Table 6. Environmental and transmitter data used in figures 18 and 19.

Profile 1 Height (ft)	Profile 1 M-units	Profile 2 Height (ft)	Profile 2 M-units
0.0	335.0	0.0	335.0
1400.0	321.0	1200.0	323.0
3500.0	396.0	3500.0	405.0
3800.0	407.0	3800.0	416.0
6000.0	486.0	7000.0	531.0
7300.0	473.0	7800.0	523.0
8000.0	498.0	8000.0	530.0

Profile 3 Height (ft)	Profile 3 M-units	Profile 4 Height (ft)	Profile 4 M-units
0.0	335.0	0.0	335.0
1000.0	325.0	700.0	328.0
3500.0	415.0	3000.0	410.0
3800.0	412.0	4000.0	400.0
6500.0	509.0	6000.0	472.0
7800.0	523.0	7500.0	457.0
8000.0	503.0	8000.0	475.0

Profile 5 Height (ft)	Profile 5 M-units
0.0	335.0
400.0	331.0
2500.0	406.0
4500.0	386.0
7000.0	476.0
7500.0	471.0
8000.0	489.0

Transmitter height = 800 and 7000 ft
Antenna beamwidth = 0.5 and 1.5
Antenna elevation angle = 0 and 0.5
Distance between profiles = 50 nmi

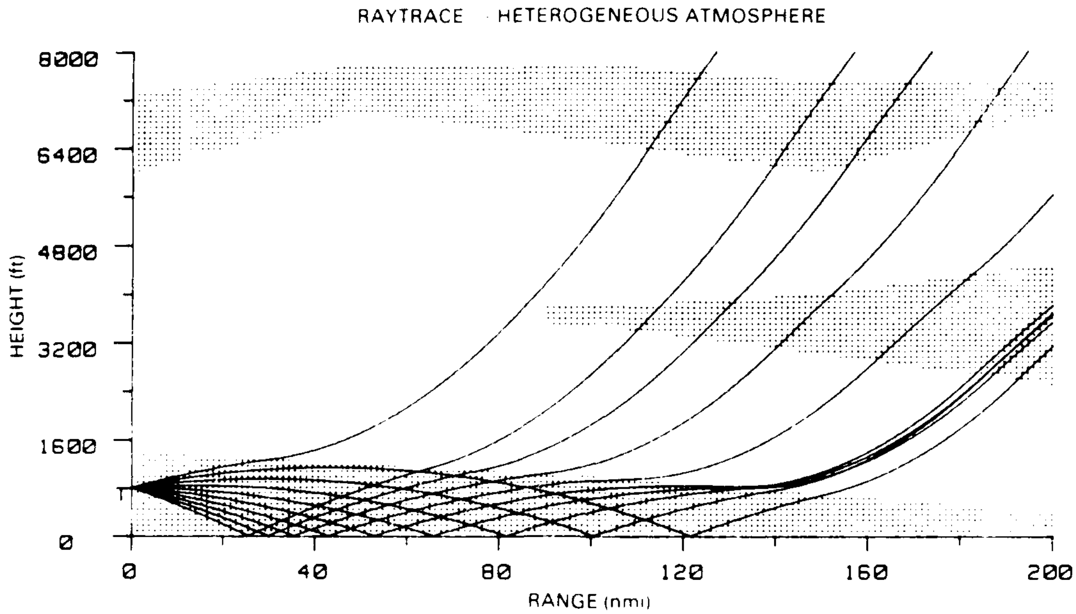


Figure 18. Ray family within a surface-based trapping layer of decreasing thickness.

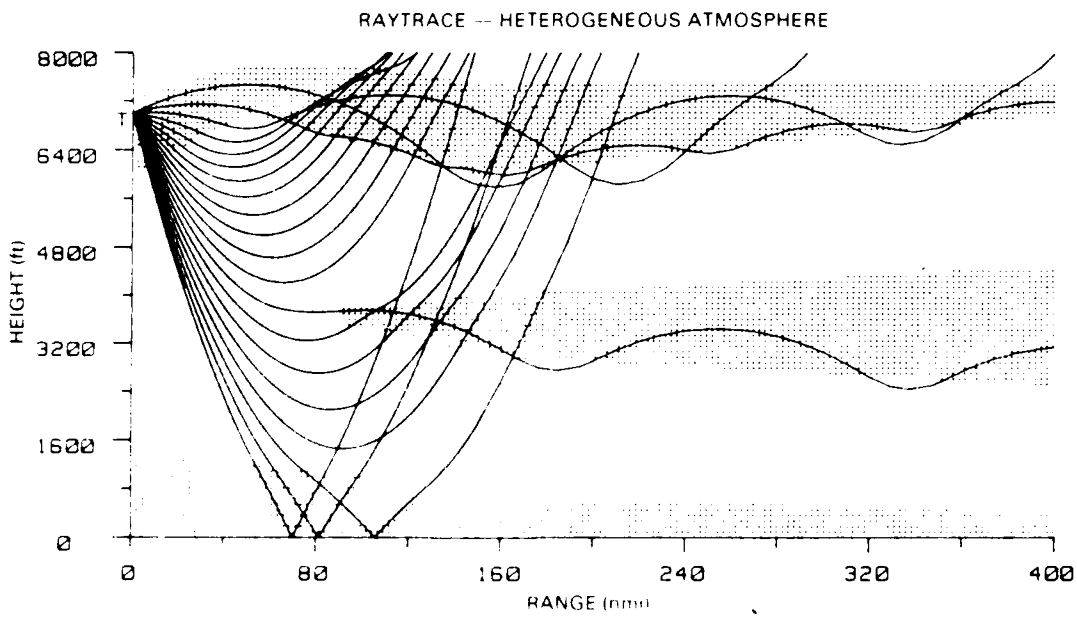


Figure 19. Ray family within an elevated trapping layer of variable thickness

REFERENCES

Bean, B. R. and E. J. Dutton, 1968. *Radio Meteorology*, Dover Publications Inc., New York

Berry, F. A., E. Bollay, and N. R. Beers, 1945. *Handbook of Meteorology*, McGraw Hill Book Company, Inc., New York.

Guinard, N. W., M. B. Lang, and K. W. Morin, June 22, 1965. "Ray Tracing in Rising and Falling Ducts," U.S. Naval Research Laboratory Report 6253.

Hitney, H. V., J. H. Richter, R. A. Pappert, K. D. Anderson, and G. B. Baumgartner, February 1985. "Tropospheric Radio Propagation Assessment," *Proceedings of the IEEE*, Vol. 73, No. 2.

Morrissey, J. F., Y. Izumi, and O. Cote, November 22, 1985. "Meteorological Measurments on Line-of-Sight Microwave Radio Links," U.S. Air Force Geophysics Laboratory Report 85-0299.

Reed, H. R. and C. M. Russell, 1966. *Ultra High Frequency Propagation*, Boston Technical Publishers, Inc., Cambridge, Massachusetts.

APPENDIX A.

Microsoft QuickBASIC (version 2.01) program code for use on IBM compatible PC, XT or AT desktop computer with MS-DOS Version 3.2.

```
'       Program:   Laterally Heterogeneous Atmosphere Raytrace
'
'       Purpose:   Trace a ray through an atmosphere while allowing the
'                  refractive index to vary both in the horizontal and
'                  the vertical.
'
'       Origin:    Naval Ocean Systems Center
'                  Code 542
'                  San Diego, CA 92152-5000
'                    Wayne L. Patterson
'
'       Edition:   13 April 1987
'
'       Glossary:
'              a        - array index counter
'              a        - intermediate variable
'              a$       - temporary string variable
'              angleinc - increment of beamwidth (radians)
'              b        - intermediate variable
'              balpha - beginning angle (radians)
'              beamwidth - antenna beamwidth
'              bht      - beginning height (meters)
'              brng     - beginning range (kilometers)
'              btmht    - height of layer's bottom (meters)
'              c        - intermediate variable
'              ch       - slope of the layer boundary
'              charwd - number of screen pixels in one character width
'              charht - number of screen pixels in one character height
'              cm       - slope of the M-unit gradient
'              col      - screen column
'              cursor - array which holds the image of a plus sign
'              delh     - change of height (meters)
'              delr     - change of range (kilometers)
'              delu     - change of M-units
'              diff     - difference between two values
'              dots     - holder of refractive condition image
'              dummy$ - temporary holder for operator inputs
'              e        - vapor pressure (millibars)
'              ealpha - ending angle (radians)
'              eht      - ending height (meters)
'              elevangle - antenna elevation angle
'              erng     - ending range (kilometers)
'              error$ - conditional checking flag
'              first    - flag to signal previous use of program
'              flag     - temporary counter
'              fttom    - feet to meters conversion factor
'              h        - temporary height (various units)
'              hite0    - height at level 0
'              hitel    - height at level 1
'              hlim     - height of tallest profile (meters)
'              hmax     - maximum height of trace (units of choice)
'              ht       - temporary height
'              htick    - number of vertical axis tick marks
'              htop     - temporary height (meters)
```

```
'               hzero  - radiosonde launch height
'               i      - array index counter
'               j      - array index counter
'               k      - array index counter
'               layer  - temporary layer counter
'               lft    - viewport's left screen column
'               lowest - lowest height within a profile (meters or
'                          pixels)
'               lower  - temporary height (meters)
'               m      - array index counter
'               matkm  - M-unit value at height of one kilometer
'               matsfc - M-unit value at the surface
'               mcolor - color to use for line drawing
'               mgrad  - M-unit gradient
'               mitokm - miles to kilometers conversion factor
'               mmax   - maximum M-unit value within a profile
'               mmin   - minimum M-unit value within a profile
'               mode$  - flag for screen type
'               most   - maximum number of levels within all profiles
'               n      - N-units
'               n      - array index counter
'               ncol   - temporary screen column counter
'               nmax   - number of levels within each environmental
'                          profile
'               nnn    - dimension of array "dots"
'               nprof  - number of profiles
'               nrays  - number of rays to trace
'               nrow   - temporary screen row counter
'               offset - incremental summation of rlim
'               ok$    - conditional checking flag
'               op     - option number
'               p      - environmental profiles
'                          index 1 = counter of profile
'                          index 2 = counter of levels
'                          index 3 = environmental data
'                              1 = heights (meters)
'                              2 = M-units
'                              3 = M-unit gradient
'               pix    - same a p but data given in screen pixels
'               pr     - array to hold profile numbers for screen
'                          prompts
'               pres   - pressure (millibars)
'               pres0  - pressure at level 0
'               pres1  - pressure at level 1
'               putdots$ - conditonal checking flag
'               pzero  - pressure at hzero
'               rh     - relative humidity (percent)
'               rlim   - range between profiles
'               rngnow - temporary range (kilometers)
'               rnginc - increment of range (kilometers)
'               row    - screen row
'               startangle - ray's initial launch angle (radians)
'               style% - style to use for line drawing
'               t      - temporary holder of environmental data
```

```
'            ta      - temperature (degrees Kelvin)
'            table   - table of tick mark values and multiplication
'                        factors
'            temp    - temperature (degrees Celsius)
'            title$  - title label
'            tlevel  - layer containing the transmitter
'            tmax    - level counter for array t (corresponds to nmax
'                         for array p)
'            topht   - height of layer's top (meters)
'            tran    - transmitter height (meters)
'            tstr0   - temperature at level 0
'            tstr1   - temperature at level 1
'            units$  - units flag for English or metric
'            upper   - temporary height (meters)
'            x       - temporary range (km, nmi, or pixels)
'            x1      - temporary range (km, nmi, or pixels)
'            xcol    - screen column counter
'            xinc    - increment of range (kilometers)
'            xlabel$ - horizontal axis labels
'            xlim    - maximum range of trace (kilometers)
'            xlimit  - temporary range limit (kilometers)
'            xmax    - maximum range of trace (units of choice)
'            xpixel  - temporary horizontal screen pixel
'            xtick   - number of horizontal axis tick marks
'            y       - temporary height (m, ft, or pixels)
'            y1      - temporary height (m, ft, or pixels)
'            ybtm    - temporary height (pixels)
'            yinc    - increment of height (meters)
'            ylabel$ - vertical axis labels
'            ylimit  - temporary height limit (meters)
'            ypixel  - temporary vertical screen pixel
'            yrow    - screen row counter
'            ytop    - temporary height (pixels)
```

```
option base 1
dim   cursor#(26), nmax(5), p(5,150,3), rlim(5), pix(5,130,3), s(5)
dim   t(5,150,2)

common shared /environment/  nmax(), nprof, p(), units$,_
                             fttom, mitokm, t()

common shared /tracecnst/    beamwidth, elevangle, hlim,_
                             nrays, rlim(), tran

common shared /graphcnst/    charwd, charht, cursor#(), mode$,_
                             pix(), x, y, hmax, htick,_
                             xmax, xtick, s(), putdots$, xcol, yrow

common shared /errorhandel/  dummy$

nnn = 15000
dim    shared  dots(nnn)

'     determine what type screen the compute is using and set
'     screen type and character size accordingly

       charwd = 8
       charht = 14

on error goto cga:  screen 9
on error goto ega:  color 7
       mode$ = "EGA16"
       goto endset

cga:   mode$ = "CGA"
       screen 2
       charht = 8
       resume endset

ega:   mode$ = "EGA4"
       resume endset

endset:  on error goto errorhandler

print
print
print
print
print "         This  raytrace program  will allow the refractive"
print "         index to vary horizontally and vertically.  Up to"
print "         5 radiosonde height versus M-unit profiles may be"
print "         used to more adequately describe the horizontally"
print "         varying atmosphere."
print
print "         For  further information  contact W.L. Patterson,"
print "         Naval Ocean Systems  Center, Code 543, San Diego,"
print "         CA, 92152-5000: Commercial 619-225-7247:  Autovon"
print "         933-7247."
```

```
print
print

locate 25,1
line input "Press <ENTER> to continue. ";ok$
cls
ok$ = "yes"
first = 0
putdots$ = "no"

while ok$ = "yes"
   if first = 0 then
      call envidata
      cls
      call mgradient
      call drawprof
      call sysdata
      call drawrange
      call raytrace
      first = 1
      putdots$ = "yes"
   else
      locate 25,1
      line input "Do you want to change environmental data? (yes,no) ",
                  ok$
      if left$(ok$,1) = "y" or left$(ok$,1) = "Y" then
          putdots$ = "no"
          call envidata
          cls
          call mgradient
          call drawprof
          call sysdata
          call drawrange
          call raytrace
          putdots$ = "yes"
      else
          call sysdata
          call drawrange
          call raytrace
      end if
   end if
   locate 25,1
   line input "Do you want to rerun the program? (yes,no)"; ok$
   if left$(ok$,1) = "y" or left$(ok$,1) = "Y" then
      window
      view
      cls
      ok$ = "yes"
   end if
wend
end

'     subroutine which will move the graphics cursor by the operator
'     holding the arrow keys
```

```
downcursor:
    put (x,y), cursor#, xor
    y = y + 1
    if y > 13 * charht then y = 1
    put (x,y), cursor#, xor
    return
leftcursor:
    put (x,y), cursor#, xor
    x = x - 5
    if x < 8 then x = 623
    put (x,y), cursor#, xor
    return
rightcursor:
    put (x,y), cursor#, xor
    x = x + 5
    if x > 623 then x = 8
    put (x,y), cursor#, xor
    return
upcursor:
    put (x,y), cursor#, xor
    y = y - 1
    if y <0.5 * charht then y = 13 * charht
    put (x,y), cursor#, xor
    return

'      error handing subroutines

errorhandler:
   if err = 75 then          ' directory all ready exists
      resume next
   elseif err = 53 then      ' file not found
      close #1
      beep
      locate 24,1
      print "File not found.  Please check spelling and try again."
      locate 25,1
      line input "Enter the desired file name. ";dummy$
      if left$(dummy$,3) = "new" or left$(dummy$,3) = "NEW" then_
          call newdata
      resume 0
   end if
   on error goto 0
```

```
'       Subroutine:   envidata

'       Purpose:  1.  Call for entry of environmental data from
                        keyboard or read the data from a previously
                        created disk file.
'                 2.  Determine the x and y axis tick mark intervals
                        based upon the maximum range and height of
                        the data.
'                 3.  Insure that all profiles start at a height of
                        zero and end at the same height

sub envidata static

dim table(7,2)

'      create an environmental files subdirectory if one does
'      not all ready exist

cls 0
mkdir "enviros"

start:
print "The below listed environmental files may be  used  for  data"
print "input by typing the file name in response to the prompt.  If"
print "new data is desired, respond to the prompt with <new>."
print
print

files "enviros\"
locate 25,1
line input "Enter the desired file name. "; dummy$
if left$(dummy$,3) = "new" or left$(dummy$,3) = "NEW" then
    call newdata
else
    open "enviros\" + dummy$ for input as #1
    dummy$ = input$(427,#1)
    units$ = input$(3,#1)
    dummy$ = input$(73,#1)
    input #1, nprof
    dummy$ = input$(21,#1)
    fttom = 0.3048
    mitokm = 1.854
    if units$ = "mks" then
         fttom = 1
         mitokm = 1
    end if

    for i = 1 to 5              ' read number of levels
        input #1, nmax(i)
    next i
    dummy$ = input$(39,#1)

    for i = 1 to 4              ' read distance between profiles
        input #1, rlim(i)
```

```
        rlim(i) = rlim(i) * mitokm
    next i
    dummy$ = input$(200,#1)

    most = 0
    for i = 1 to nprof          ' determine the most number of levels
        if nmax(i) > most then most = nmax(i)
    next i

    for j = 1 to most           ' read the height and M-units
        for i = 1 to nprof
            input #1, p(i,j,1), p(i,j,2)
            p(i,j,1) = p(i,j,1) * fttom
        next i
        dummy$ = input$(2,#1)
    next j
    close #1
end if


'     find the highest height of all the profiles and the total
'     range

hlim = 0
xlim = 0
for i = 1 to nprof
    if p(i,nmax(i),1) > hlim then hlim = p(i,nmax(i),1)
    if i < nprof then xlim = xlim + rlim(i)
next i

'     determine the number and intervals of the vertical
'     tick mark

restore bound
bound:
data 4,4,5,5,8,4,10,5,15,5,20,4,25,5
for i = 1 to 7
    read table(i,1), table(i,2)
next i

hmax = hlim / fttom
factr = 1
aa:
    for i = 1 to 7
        if hmax <= table(i,1) * factr + 0.1 then goto bb
    next i
    factr = factr * 10
    goto aa

bb:
    hmax = table(i,1) * factr
    htick = table(i,2)

'     determine the number and intervals of the horizontal
```

```
'     tick marks

    xmax = xlim / mitokm
    factr = 1
cc:
    for i = 1 to 7
        if xmax <= table(i,1) * factr then goto dd
    next i
    factr = factr * 10
    goto cc

dd:
    xmax = table(i,1) * factr
    xtick = table(i,2)

'     compute the M-unit value at one kilometer for each profile

for i = 1 to nprof
    ok$ = "no"
    j = nmax(i)
    if p(i,j,1) < 1000 then
        diff = (1000 - p(i,j,1)) / 1000
        matkm = (118 * diff) + p(i,j,2)
    elseif p(i,j,1) = 1000 then
        matkm = p(i,j,2)
    else
        while ok$ = "no"
            if p(i,j,1) > 1000 then
                j = j - 1
            else
                matkm = p(i,j+1,2) - (p(i,j+1,2) - p(i,j,2)) *_
                        (p(i,j+1,1) - 1000) / _
                        (p(i,j+1,1) - p(i,j,1))
                ok$ = "yes"
            end if
        wend
    end if

'     if necessary, compute the M-unit value at the height (hlim)
'     and insert it into the profile

    if p(i,nmax(i),1) < hlim then
        nmax(i) = nmax(i) + 1
        p(i,nmax(i),1) = hlim
        h = hlim  / 1000
        p(i,nmax(i),2) = 157 * h + p(i,1,2) * exp(-log(p(i,1,2)_
                         / (matkm - 157)) * h)
    end if
next i
end sub
```

```
'     Subroutine:  sysdata

'     Purpose:     Solicit the transmitter data from the operator


sub sysdata static

cls 0
ok$ = "no"
while ok$ = "no"
   if units$ = "fps" then
      locate 24,1
      print "Transmitter must be between 0 and";_
              int(hlim/fttom); "feet."
      locate 25,1
      line input "Enter transmitter height (feet) "; dummy$
   else
      locate 24,1
      print "Transmitter must be between 0 and";_
             int(hlim); "meters."
      locate 25,1
      line input "Enter transmitter height (meters) "; dummy$
   end if
   tran = val(dummy$) * fttom
   if tran < 0 or tran > hlim then
      beep
      print "Transmitter height must be between 0 and ";_
            hlim / fttom;". Try again."
   else
      ok$ = "yes"
   end if
wend

cls 0
locate 25,1
ok$ = "no"
while ok$ = "no"
   input "Enter beamwidth in degrees (0.5 to 45). ", beamwidth
   if beamwidth < 0.5 or beamwidth > 45 then
      beep
      print "Beamwidth must be between 0.5 and 45.  Try again."
   else
      ok$ = "yes"
   end if
wend

cls 0
locate 25,1
ok$ = "no"
while ok$ = "no"
   input "Enter elevation angle in degrees (-10 to 10). ",_
         elevangle
   if elevangle < -10 or elevangle > 10 then
```

```
      beep
      print "Elevation angle must be between -10 and 10.  Try again."
   else
      ok$ = "yes"
   end if
wend

cls 0
locate 25, 1
ok$ = "no
while ok$ = "no"
   input "Enter number of rays to trace (1 to 100). ", nrays
   if nrays < 1 or nrays > 100 then
      beep
      print "Number of rays must be between 1 and 100.  Try again."
   else
      ok$ = "yes"
   end if
wend
cls 0
end sub
```

```
'      Subroutine:   drawprof

'      Purpose:      1.  Draw the M-unit versus height profiles on the
'                          screen
'                    2.  Determine if layers can be automaticly defined.
'                          If not, call for layer drawing subroutine.

sub drawprof static

'      print the graph title and vertical axis label

locate 1,21
print "SELECTED HEIGHT VERSUS M UNIT PROFILES"
a$ = "HEIGHT"
row = 9
for i = 1 to 6
    locate row,1
    print mid$(a$,i,1)
    row = row + 1
next i

'      define the left margin (column number) of each drawing cell

restore bounds
bounds: data 3, 19, 35, 51, 67
col = 6

'      loop to draw each profile cell

for i = 1 to nprof

'      print cell title and horizontal axis label

    locate 3,col-1
    print "PROFILE" + str$(i)
    locate 20,col
    print "M UNITS"
    col = col + 16

'      open the viewport and find the mininum and maximum
'      x axis value

    read lft
    view (lft*charwd,4*charht) - ((lft+11)*charwd, 18*charht)
    mmin = 1.e5
    mmax = 0
    for j = 1 to nmax(i)
        if p(i,j,2) > mmax then mmax = p(i,j,2)
        if p(i,j,2) < mmin then mmin = p(i,j,2)
    next j

'      define the drawing window and draw the horizontal axis

    window (mmin, p(i,nmax(i),1)/ fttom) - (mmax, 0)
```

```
line (mmin,0) - (mmax,0),14

'     draw the m unit versus height profile using various colors
'     for the gradients

for j = 1 to nmax(i) - 1
    if p(i,j,3) <= 0 then
        mcolor = 4
        style% = &hffff
    elseif p(i,j,3) < 7.9e-5 then
        mcolor = 5
        style% = &hf18f
    elseif p(i,j,3) <= 1.57e-4 then
        mcolor = 2
        style% = &hcccc
    else
        mcolor = 1
        style% = &hff00
    end if

    if mode$ <> "CGA" then style% = &hffff
    line (p(i,j,2), p(i,j,1)/fttom) -_
         (p(i,j+1,2),p(i,j+1,1)/fttom),_
          mcolor, , style%

'     fill the pix array with x axis pixels, y axis heights
'     and z axis 1.e7

     pix(i,j+1,1) = point(0)+lft*charwd
     pix(i,j+1,2) = point(1)
     if p(i,j+1,1) = 0 then pix(i,j+1,3) = 1.e7
     if p(i,j+1,1) <> 0 then  pix(i,j+1,3) =  p(i,j+1,1)
     next j
next i

window
view
row = 22
call refcond(row)

'     check to see if all profiles have same number of points

ok$ = "yes"
i = 1
while ok$ = "yes" and i <= nprof
    if nmax(1) <> nmax(i) then
        ok$ = "no"
        flag = -1
    else
        i = i + 1
    end if
wend

'     if all profiles have same number of points, count number of
```

```
'     levels on each profile that has a negative m-unit gradient

if ok$ = "yes" then
    i = 1
    while i <= nprof
        flag = 0
        j = 1
        while  j <= nmax(1)
            if p(i,j,3) < 0 then flag = flag + 1
            j = j + 1
        wend
        i = i + 1
    wend
end if

'     draw boundary lines and ask for approval if equal number of
'     points otherwise go straight to operator drawing routine

if ok$ = "yes" then
    window
    view (1, 4*charht) - (639, 18*charht)
    mcolor = 4
    for j = 2 to nmax(1)
        for i = 1 to nprof - 1
            if mode$ = "EGA16" then
                 if p(i,j,3) <= 0 then
                     mcolor = 4
                 elseif p(i,j,3) < 7.9e-5 then
                     mcolor = 5
                 elseif p(i,j,3) <= 1.57e-4 then
                     mcolor = 2
                 else
                     mcolor = 1
                 end if
            end if

            line (pix(i,j,1),pix(i,j,2)) - (pix(i+1,j,1),_
                  pix(i+1,j,2)), mcolor, , &hffff
        next i
    next j
    locate 23,1
    beep
    if flag = 0 then
        print "Note!  Trapping layer not continuous between profiles."
    else
        print "Note!  Multiple trapping layers within a profile."
    end if

    line input; "Want to  define the layers yourself?  (yes or no) ";_
                 ok$
    if left$(ok$,1) = "y" or left$(ok$,1) = "Y" then

'     erase the boundary line previously drawn and call for drawing
'     routine
```

```
        for j = 2 to nmax(1)
              for i = 1 to nprof - 1
                  line (pix(i,j,1),pix(i,j,2)) - (pix(i+1,j,1),_
                       pix(i+1,j,2)), 0, , &hffff
              next i
          next j
          window
          view
          call drawlayers
       end if
   else
       window
       view
       beep
       if flag = - 1 then
           locate 23,1
           print "Profiles contains unequal number of levels."
       end if
       locate 24,1
       line input; "Press <ENTER> to continue. "; ok$
       call drawlayers
   end if

   cls 0
   end sub
```

```
'       Subroutine:  drawlayers

'       Purpose:      1.  Prompt the operator for determination of layer
'                         boundaries.
'                     2.  Re-establish the profiles by including any new
'                         layers as  defined by the operator

sub drawlayers static

'       fill the prompt array.  These prompts are profile numbers which
'       will be considered next.

dim pr(5,5,5), tmax(150)
restore a
for i = 1 to 5
    for j = 1 to 5
        for k = 1 to 5
            read pr(i,j,k)
        next k
    next j
next i

a:
data 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
data 2,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
data 2,3,0,0,0,3,2,1,0,0,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0
data 2,3,4,0,0,3,4,3,1,0,4,2,2,1,0,3,2,1,0,0,0,0,0,0,0
data 2,3,4,5,0,3,4,5,4,1,4,5,3,2,1,5,2,3,2,1,4,3,2,1,0

'       fill the first layer of the temporary profile array
'       with the first layer of the environmental array

for i = 1 to nprof
    t(i,1,1) = p(i,1,1)
    t(i,1,2) = p(i,1,2)
    tmax(i) = 1
next i

'       define the graphics crosshair and save the image

view(1, 21 * charht) - (639, 23 * charht)
cls
view
locate 2,2
line (1, 0.5* charht) - (2 * charwd,0.5 * charht), 14
line (charwd, 1) - (charwd, charht), 14
get (1, 1) - (2 * charwd, charht), cursor#
put (1, 1), cursor#, xor

view (1, 4 * charht) - (639, 18 * charht)

'       find the most number of points in any profile

most = 0
```

```
for i = 1 to nprof
    if nmax(i) > most then most = nmax(i)
next i

'      start the loop for solicitation of points

ok$ = "yes"
layer = 2
while ok$ <> "no"

'      find a starting point - the lowest nonused point of all profiles

    ok$ = "no"
    j = 2
    lowest = 0
    while ok$ = "no" and j <= most
        for i = 1 to nprof
            if pix(i,j,3) <> 1.e7 and pix(i,j,2) > lowest then
                n = i
                jj = j
                lowest = pix(i,j,2)
                ok$ = "yes"
            end if
        next i
        if ok$ = "no" then j = j + 1
    wend

    if ok$ = "yes" then
        k = 1
        for j = 1 to nprof -1

'      put the starting point into the temporary profile array
'      and flag it as used

            t(n,layer,1) = p(n,jj,1)
            t(n,layer,2) = p(n,jj,2)
            pix(n,jj,3) = 1.e7

'      put the crosshair at the starting point

            if j = 1 or k = pr(nprof,n,k) then

'      the next if statement is a special case fix

                if nprof = 4 and n = 4 and k = 2 then goto 10
                x = pix(n,jj,1) - charwd
                y = pix(n,jj,2) 0.5 * charht
                x1 = pix(n,jj,1)
                y1 = pix(n,jj,2)
                put (x,y), cursor#, xor
                if j <> 1 then k = k + 1
            end if
10
            n = pr(nprof,n,k)
```

```
        error$ = "yes"
        while error$ = "yes"

'     solicit for next point

            locate 24,1
            beep
            print "Using arrow keys, select point from profile";_
                  pr(nprof,n,k); "then press ENTER. ";

'     turn on the arrow keys and event trap

            key (11) on
            key (12) on
            key (13) on
            key (14) on
            on key (11) gosub upcursor
            on key (12) gosub leftcursor
            on key (13) gosub rightcursor
            on key (14) gosub downcursor

incursor:
            a$ = ""
            while a$ = ""
                a$ = inkey$
            wend
            if a$ = chr$(13) then goto nextprof
            goto incursor

nextprof:

'      convert the input y pixel to height

            h = y + 0.5 * charht
            ytop = 1
            ybtm = 14 * charht
            htop = p(m,nmax(m),1)
            h = htop - htop * (ytop-h) / (ytop-ybtm)

'     find the lowest nonused point (a) on the next profile

            a = 2
            for i = 1 to nmax(m)
                if pix(m,i,3) = 1.e7 then a = a + 1
            next i

'     compute acceptable bounds about the point "a"

            upper = p(m,a,1) + p(m,a,1) * 0.05
            lower = p(m,a,1) - p(m,a,1) * 0.05

'     check the input height's relationship to the bounds

            if h < lower then
```

```
                    if h > t(m,layer-1,1) then

'       input height less then bottom bound and greater than the last
'       temporary profile point so interpolate for M-unit value and
'       insert the new level into the temporary profile array

                        t(m,layer,2) = p(m,a,2) - ((p(m,a,2) -_
                                       p(m,a-1,2)) * (p(m,a,1) - h)_
                                       / (p(m,a,1) - p(m,a-1,1)))
                        t(m,layer,1) = h
                        tmax(m) = tmax(m) + 1
                        tmax(n) = tmax(m)
                        error$ = "no"
                    else

'       input height less than previous temporary profile point so give
'       crossing bounds error message

                        beep
                        locate 23,1
                        print "Boundaries can't cross.  Press ENTER.";

pausea:
                        a$ = ""
                        while a$ = ""
                            a$ = inkey$
                        wend
                        if a$ <> chr$(13) then goto pausea
                        locate 23,1
                        print
                    end if
                elseif h > upper then

'       input height greater than bounds so give missed breakpoint error
'       message

                    beep
                    locate 23,1
                    print "M-unit breakpoint exceeded.  Press ENTER.";

pauseb:
                    a$ = ""
                    while a$ = ""
                        a$ = inkey$
                    wend
                    if a$ <> chr$(13) then goto pauseb
                    locate 23,1
                    print
                else

'       input height within bounds so insert the point (a) into the
'       temporary profile array and flag it as being used

                    t(m,layer,1) = p(m,a,1)
```

```
                        t(m,layer,2) = p(m,a,2)
                        tmax(m) = tmax(m) + 1
                        tmax(n) = tmax(m)
                        pix(m,a,3) = 1.e7
                        error$ = "no"
                    end if
                    if error$ = "yes" then

'     if there was an error, move crosshairs back to starting point

                        put (x,y), cursor#, xor
                        x = x1 - charwd
                        y = y1 - 0.5 * charht
                        put (x,y), cursor#, xor
                    else

'     if no error, draw boundary line between the two profiles
'     erase the crosshairs if on the last profile and move on
'     to the next profile

                        line (x1, y1) - (x + charwd, y + 0.5 * charht), 14
                        x1 = x + charwd
                        y1 = y + 0.5 * charht
                        if j = nprof -1 then
                            put (x,y), cursor#, xor
                        elseif j = nprof - n then
                            put (x,y), cursor#, xor
                        end if
                        k = k + 1
                    end if
                wend
            next j
            layer = layer + 1
        end if
    wend

'     refill the profile array with the temporary height M-unit values
'     and recompute the M-unit gradients

    for i = 1 to nprof
        tmax(i) = tmax(i) + 1
        t(i,tmax(i),1) = p(i,nmax(i),1)
        t(i,tmax(i),2) = p(i,nmax(i),2)
    next i

    for i = 1 to nprof
        nmax(i) = tmax(i)
        for j = 1 to nmax(i)
            p(i,j,1) = t(i,j,1)
            p(i,j,2) = t(i,j,2)
        next j
    next i
    call mgradient
    end sub
```

```
'     Subroutine:  Drawrange

'     Purpose:      1. Draw the axis and labels for the raytrace

sub drawrange static

xlabel$ = "Range in Nautical Miles"
ylabel$ = "Height in Feet"
title$ = "Raytrace - Laterally Heterogeneous Atmosphere"
if units$ = "mks" then
    xlabel$ = "Range in Kilometers"
    ylabel$ = "Height in Meters"
end if

'     establish drawing limits and tick mark increments

xlimit = xmax
ylimit = hmax
xinc = xlimit / xtick
yinc = ylimit / htick

'     determine the start points based upon the terminal screen being
'     used and draw the axis

view
if htick = 4 then
    yrow = 18.5
    nrow = 4
else
    yrow = 17.5
    nrow = 3
end if
if xtick = 4 then
    xcol = 76.5
    ncol = 16
else
    xcol = 72.5
    ncol = 12
end if

line (12.5 * charwd, 2.5 * charht) - (12.5 * charwd, yrow * charht),7
line (12.5 * charwd, yrow * charht) - (xcol * charwd, yrow * charht),7
ypixel = 2.5 * charht: xpixel = 10 * charwd

'     draw and label the vertical axis tick marks

row = 3
fac = htick
for i = 2 * htick to 0 step -1
    if i mod 2 = 0 then
        line (xpixel,ypixel) - (xpixel + 2.5 * charwd, ypixel)
        locate row, 4
        print yinc * fac
        fac = fac - 1
```

```
        row = row + nrow
    else
        line (xpixel+charwd, ypixel) - (xpixel+2.5*charwd, ypixel)
    end if
    ypixel = ypixel + nrow * charht / 2
next i

'     draw and label the horizontal axis tick marks

col = 12
fac = 0
ypixel = yrow * charht
xpixel = xpixel + 2.5 * charwd
for i = 0 to 2 * xtick
    if i mod 2 = 0 then
        line (xpixel, ypixel) - (xpixel, ypixel + charht)
        locate 20, col
        print xinc * fac
        fac = fac + 1
        col = col + ncol
    else
       line (xpixel, ypixel) - (xpixel, ypixel + 0.5 * charht)
    end if
    xpixel = xpixel + ncol * charwd / 2
next i

'     print the title and horizontal axis labels

locate 1,  (80 - len(title$))  / 2: print title$
locate 21, (80 - len(xlabel$)) / 2: print xlabel$

'     print the vertical axis label

for i = 1 to len(ylabel$)
    locate (20 - len(ylabel$)) / 2 + i, 2
    print mid$(ylabel$,i,1)
next i

'     if screen is enhanced graphics with extra memory, call for
'     refractive conditions legion

row = 22
if mode$ = "EGA16" then call refcond(row)

'     define the view port and window for the raytrace graph

view (12.5 * charwd, 2.4 * charht) - (xcol * charwd, yrow * charht)
window (0,hmax) - (xmax,0)
end sub
```

```
'       Subroutine:   refcond

'       Purpose:      1.  Draw the refractive condition legion at the
'                         bottom of the screen

sub refcond(row) static

locate row,10
print "TRAPPING"
locate row,27
print "SUPERREFRACTIVE"
locate row,51
print "STANDARD"
locate row,68
print "SUBREFRACTIVE"

'       define the box boundaries, interior colors and line style

restore limits
limits: data 3, 4, &hffff, 20, 5, &hf18f, 44, 2, &hcccc, 61, 1, &hff00

'       read the drawing attributes, draw and fill the box

for i = 1 to 4
    read lft, clr, style%
    if mode$ = "EGA16" then
        view screen (lft*charwd, (row-1)*charht) - _
                    ((lft+5)*charwd, (row)*charht)
        paint ((lft+1)*charwd, (row-0.5)*charht), clr, clr
    else
        line (lft*charwd, row*charht - 0.5*charht) -
             ((lft+5)*charwd, row*charht -0.5*charht), clr,  , style%
    end if
next i
end sub
```

```
'     Subroutine:  newdata

'     Purpose:     1.  Solicit the operator for environemntal data.
'
'                  2.  If data entered in units other than M-units,
'                        convert to M-units

sub newdata static
dim  e(31), pres(31), rh(31), table(7,2), temp(31)

'     initialize  arrays and constants

erase e nmax p pres rh rlim temp
units$ -"fps"
fttom - 0.3048
mitokm - 1.854
cls

locate 4,1
print
print    "Atmosphere specification options are:"
print
print   "    1.  Pressure, temperature, and relative humidity"
print   "    2.  Height and N-units"
print   "    3.  Height and M-units"
ok$ - "no"
while ok$ - "no"
    locate 25,1
    input; "Enter specification option (1, 2 or 3)? ", op
    if op < 1 or op > 3 then
        beep
        print "Option must be between 1 and 3.  Try again."
    else
        ok$ - "yes"
    end if
wend

cls
locate 25,1
line input; "Enter units of height/range (english or metric). ";_
            dummy$
if left$(dummy$,1) - "m" or left$(dummy$,1) - "M" then
    fttom - 1
    mitokm - 1
    units$ - "mks"
end if

cls
ok$ - "no"
while ok$ - "no"
    locate 25,1
    input "How many profiles do you want to enter (1 to 5)? ", nprof
    if nprof < 1 or nprof > 5 then
        beep
```

```
            print "Number of profiles must be between 1 and 5.  Try again."
        else
            ok$ = "yes"
        end if
wend

'     enter the atmosphere data

for i = 1 to nprof
    cls
    locate 1,1
    print "Data entry for profile"; i; "of"; nprof
    print
    j = 1
    ok$ = "no"
    while ok$ = "no" and j <= 30
        print "For level"; j
        if op = 1 then
            if j = 1 then
                if units$ = "fps" then
                    input "Enter radiosonde launch ht (ft), pres (mb)_
temp (C) and rh(%). ", hzero, pres(j), temp(j), rh(j)
                else
                    input "Enter radiosonde launch ht (meters), pres_
(mb), temp(C) and rh(%). ",hzero, pres(j), temp(j), rh(j)
                end if
                p(i,j,1) = hzero * fttom
                nmax(i) = nmax(i) + 1
                j = j + 1
            else
                input "Enter pressure (mb), temp (C), and rh (-1, -1,_
-1 to end) ", pres(j), temp(j), rh(j)
                if pres(j) < 0 then
                    ok$ = "yes"
                elseif pres(j) >= pres(j-1) then
                    beep
                    print "Pressure must decrease.  Try again."
                else
                    nmax(i) = nmax(i) + 1
                    j = j + 1
                end if
            end if
        else
            if units$ = "mks" then
                if op = 2 then
                    input "Enter height (meters) and N units (-1,-1 to_
end) ", ht, n
                    m = n + ht/6.371
                else
                    input "Enter height (meters) and M units (-1,-1 to_
 end) ", ht, m
                end if
            else
                if op = 2 then
```

```
                    input "Enter height (feet) and N-units (-1, -1 to_
end) ", ht, n
                    m = n + ht*fttom/6.371
                else
                    input "Enter height (feet) and M-units (-1.-1 to _
end) ", ht, m
                end if
            end if
            ht = ht * fttom
            if ht >= 0 then
                if j = 1 then
                    p(i,j,1) = ht
                    p(i,j,2) = m
                    nmax(i) = j
                    j = j + 1
                else
                    if ht <= p(i,j-1,1) then
                        beep
                        print "Heights must be in increasing order.  _
Try again."
                    else
                        p(i,j,1) = ht
                        p(i,j,2) = m
                        nmax(i) = j
                        j = j + 1
                    end if
                end if
            else
                ok$ = "yes"
            end if
        end if
    wend

'     if pressure entered, convert to height and compute M-units

    if op = 1 then
        hite0 = hzero
        pres0 = pres(1)
        tstr0 = temp(1) + 273.2
        for j = 1 to nmax(i)
            ta = temp(j) + 273.2
            pres1 = pres(j)
            ee = 6.105 * exp(25.22 * (ta-273.2)/ta - 5.31 * _
                 log(ta/273.2)) * rh(j)/100
            tstr1 = ta + .3794017 * ta * ee/ (pres1 - ee)
            hite1 = hite0 + 14.643 * (tstr1 + tstr0) * _
                    log(pres0/pres1)
            p(i,j,1) = hite1
            p(i,j,2) = 77.6/ta * (pres1 + 4810 * ee/ta) +_
                       hite1/6.371
            hite0 = hite1
            pres0 = pres1
            tstr0 = tstr1
        next j
```

```
    end if

'       if necessary, set surface height to zero and compute a M-unit
'       value

    if p(i,1,1) <> 0 then
        for j = nmax(i) + 1 to 2 step -1
            p(i,j,1) = p(i,j-1,1)
            p(i,j,2) = p(i,j-1,2)
        next j
        p(i,1,1) = 0
        p(i,1,2) = p(i,1,2) - 118 * p(i,2,1)/1000
        nmax(i) = nmax(i) + 1
    end if

    cls
    locate 25,1
    if nprof = 1 then
        if units$ = "fps" then
            input "Enter range (nautical miles) for raytrace. ",_
                  rlim(i)
        else
            input "Enter range (kilometers) for raytrace. ", rlim(i)
        end if
        rlim(i) = rlim(i) * mitokm
        for j = 1 to nmax(i)
            p(2,j,1) = p(1,j,1)
            p(2,j,2) = p(1,j,2)
        next j
        nmax(2) = nmax(i)
        nprof = 2
    elseif i < nprof then
        if units$ = "fps" then
            input "Enter range (nautical miles) to next profile. ",_
                   rlim(i)
        else
            input "Enter range (kilometers) to next profile. ",_
                   rlim(i)
        end if
        rlim(i) = rlim(i) * mitokm
    end if
next i

'       solicit the operator for a storage file name if desired

cls
ok$ = "no"
while ok$ = "no"
    locate 25,1
    line input; "Do you want to store this data for future use _
(yes,no)?", dummy$
    if left$(dummy$,1) = "y" or left$(dummy$,1) = "Y" then

        cls
```

```
        locate 25,1
        line input; "Enter file name for storage of data (8 characters_
 max). "; dummy$
        open "enviros\" + dummy$ for output as #1

        print #1, "This file contains environmental data for the multi_
-profile"
        print #1, "raytrace. If you edit this file, the colons (:) mus_
t remain"
        print #1, "in  place.  All  numbers  must be separated by at l_
east one"
        print #1, "space. The column structure of the heights and M-un_
its must"
        print #1, "be retained.  Heights must be in increasing order. _
There is"
        print #1, "no error checking on a stored file.                _
"
        print #1, "                                                   _
"
        print #1, using "\ \"; units$;
        print #1, using "&";"                  :units of height/range._
 mks = metric.  fps = English."
        print #1, using "#"; nprof;
        print #1, using "&";"                    :number of profiles  "

        most = 0
        for i = 1 to 5
            print #1, using "###"; nmax(i);
            if nmax(i) > most then most = nmax(i)
        next i
        print #1, using "&";"      :number of levels within each profi_
le"

        for i = 1 to 4
            print #1, using "#####"; rlim(i) / mitokm;
        next i
        print #1, using "&";" :distance between profiles in units of a_
bove"

        print #1, ""
        print #1, "   Profile 1     Profile 2     Profile 3      Prof_
le 4     Profile 5"
        print #1, " height M-unit  height M-unit  height M-unit  height_
t M-unit  height M-unit"
        print #1, ""

        for j = 1 to most
            for i = 1 to nprof
                print #1, using "#####.#"; p(i,j,1) / fttom;
                print #1, using " ####.# "; p(i,j,2);
            next i
            print #1, chr$(13)
        next j
        close #1
```

```
          ok$ = "yes"
      elseif left$(dummy$,1) ="n" or left$(dummy$,1) = "N" then
          ok$ = "yes"
      else
         beep
         print "Response must be yes or no.  Try again."
      end if
   wend
   end sub
```

```
'      Subroutine:  raytrace

'      Purpose:     1.  By taking a range step, compute an ending
'                       height and angle for a ray with a specified
'                       launch angle.

sub raytrace static

dim   offset(5), ch(5,150), cm(5,150)

startangle = (elevangle - (0.5 * beamwidth)) * 0.0174532
angleinc = (beamwidth / nrays) * 0.0174532

'      total the range increments and compute incremental range offsets

xlimit = 0
for i = 1 to nprof - 1
    xlimit = xlimit + rlim(i)
next i

offset(1) = 0
offset(2) = rlim(1)
offset(3) = offset(2) + rlim(2)
offset(4) = offset(3) + rlim(3)
offset(5) = offset(4) + rlim(4)

'      compute the layer height and M-unit gradient coefficients

for i = 1 to nprof - 1
    for j = 1 to nmax(i)
        ch(i,j) = (p(i+1,j,1) - p(i,j,1)) / rlim(i)
        cm(i,j) = (p(i+1,j,3) - p(i,j,3)) / rlim(i)
    next j
next i

'      ask for display of refractive conditons

if putdots$ = "no" then
    locate 25,1
    if mode$ = "EGA16" then
        line input; "Want to see all conditions or just trapping (all _
or trap)?           ",dummy$
    else
        print "                                                          _
          ";
    end if

'      draw dots for refractive conditions

    locate 25,1
    if mode$ <> "EGA16" then
        print "Drawing trapping regions.  Please standby.                _
            ";
    else
```

```
          print "
              ";
      end if

      if xtick = 4 then xinc = xlimit / 128
      if xtick = 5 then xinc = xlimit / 120
      if mode$ <> "CGA" then
          if htick = 4 then yinc = int(hlim) / 112
          if htick = 5 then yinc = int(hlim) / 105
      else
          if htick = 4 then yinc = int(hlim) / 64
          if htick = 5 then yinc = int(hlim) / 60
      end if
      y = yinc
      i = 1
      j = 1
      n = 1
      while y < hlim - yinc
          x = xinc
          while x < xlimit
              ytop = ch(i,j+1) * (x-offset(i)) + p(i,j+1,1)
              ybtm = ch(i,j) * (x-offset(i)) + p(i,j,1)
              if y > ytop then
                  j = j + 1
              elseif y < ybtm then
                  j = j - 1
              end if
              mgrad = cm(i,j) * (x-offset(i)) + p(i,j,3)
              if mode$ = "EGA16" and (left$(dummy$,1) = "a" or _
                          left$(dummy$,1) = "A") then
                  if mgrad <= 0 then
                      mcolor = 4
                  elseif mgrad < 7.9e-5 then
                      mcolor = 5
                  elseif mgrad <= 1.57e-4 then
                      mcolor = 2
                  else
                      mcolor = 1
                  end if
                  pset (x/mitokm,y/fttom), mcolor
              else
                  if mgrad < 0 then pset (x/mitokm,y/fttom), 4
              end if
              x = x + xinc
              if x > offset(i+1) then i = i + 1
          wend
          y = y + yinc
          i = 1
          if y > p(i,n+1,1) then n = n + 1
          j = n
      wend
      locate 25,1
      if mode$ <> "EGA16" then print "
                                     ";
```

```
'     put the dots in an array for later display

    window
    view
    get (12.5*charwd+0.1*charwd, 2.4*charht+0.1*charht)_
      - (xcol*charwd-0.1*charwd, yrow*charht-0.1*charht), dots
else

    window
    view
    put (12.5*charwd+ 0.1*charwd, 2.4*charht+0.1*charht),  dots, xor
end if

'     establish viewport and window for drawing

view (12.5 * charwd, 2.4 * charht) - (xcol * charwd, yrow * charht)
window (0,hmax) - (xmax,0)

'     find the layer containing the transmitter

ok$ = "no"
i = 1
while ok$= "no " and i <= nmax(1)
    if tran < p(1,i,1) then
        i = i - 1
        ok$ = "yes"
    elseif tran = p(1,i,1) then
        ok$ = "yes"
    else
        i = i + 1
    end if
wend
tlevel = i

for j = 1 to nrays

'     initialize the beginning values

    n = 1
    i = tlevel
    bht = tran
    eht = bht
    brng = 0
    erng = 0
    rngnow = rlim(n)
    rnginc = rlim(n) / 25
    if j = 1 then
        balpha = startangle
    else
        balpha = startangle + (j -1) * angleinc
    end if

'     loop until maximum height or range is reached
```

```
    while eht < hmax / fttom and erng < xlimit and i < nmax(1)

'      compute local M-unit gradient, ending range and angle

         mgrad = cm(n,i) * (brng-offset(n)) + p(n,i,3)
         if (mgrad = 0) then mgrad = 1.e-6
         erng = brng + rnginc
         if (erng > rngnow) then erng = rngnow
         ealpha = balpha + mgrad * (erng - brng)

'      check to see if ray has passed through a maximum or minimum
'      if so, compute a new ending range and angle

         if (balpha<0 and ealpha>=0) or (balpha>0 and ealpha<=0) then
             ealpha = 0
             erng = brng - balpha / mgrad
         end if

'      compute an ending height and the boundary heights at this
'      ending range

         eht = bht + (ealpha^2 - balpha^2) / (2.e-3 * mgrad)
         topht = ch(n,i+1) * (erng-offset(n)) + p(n,i+1,1)
         btmht = ch(n,i)   * (erng-offset(n)) + p(n,i,1)

'      if ray has penetrated layer boundary, compute range, height, and
'      angle of boundary crossing

         if eht > topht or eht < btmht then
             if eht > topht then
                 ll = i + 1
                 flag = 1
             else
                 ll = i
                 flag = 0
             end if

             a = mgrad / 2.e-3
             b = 2 * balpha / 2.e-3 - ch(n,ll)
             c = bht - ch(n,ll) * (brng - offset(n)) - p(n,ll,1)
             delr = (-b + sqr(b^2 - 4 * a * c)) / (2 * a)

             if delr < 0 or delr > rnginc then_
                   delr = (-b - sqr(b^2 - 4 * a * c)) / (2 * a)
             if delr = 0 then delr = rnginc
             erng = brng + delr
             ealpha = balpha + mgrad * (erng - brng)
             eht = ch(n,ll) * (erng - offset(n)) + p(n,ll,1)
             if flag = 1 then
                 i = i + 1
             else
                 i = i - 1
             end if
        end if
```

```
'     reverse angle if ray has reached the ground

        if (eht <= 0) then
            i = 1
            ealpha = -ealpha
            eht = 0
        end if

'     draw the ray segment on the screen

        line (brng/mitokm, bht/fttom) - (erng/mitokm, eht/fttom),14

'     check to see if ray has reached next profile range.  If so,
'     increment range counter and get new step size

        if erng = rngnow and erng < xlimit then
            n = n + 1
            rngnow = rngnow + rlim(n)
            rnginc = rlim(n) / 25
        end if

'     reinitialize starting height, angle, and range

        bht = eht
        balpha = ealpha
        brng = erng
    wend
next j

'     toggle background

window
view
locate 25,1
print "Press <t> to toggle background or <ENTER> to continue.";

toggle:
   a$ = ""
   while a$ = ""
       a$ = inkey$
   wend
   if a$ = chr$(84) or a$ = chr$(116) then_
      put (12.5*charwd+ 0.1*charwd, 2.4*charht+0.1*charht),dots, xor
   if a$ = chr$(13) then goto continue
   goto toggle

continue:

end sub
```

```
'     Subroutine:  gradient

'     Purpose:     1.  Compute an M-unit gradient

sub gradient static

for i = 1 to nprof

'     compute the M-unit value at one kilometer

    ok$ = "no"
    j = nmax(i)
    if p(i,j,1) < 1000 then
        diff = (1000 - p(i,j,1)) / 1000
        matkm = (118 * diff) + p(i,j,2)
    elseif p(i,j,1) = 1000 then
        matkm = p(i,j,2)
    else
        while ok$ = "no"
            if p(i,j,1) > 1000 then
                j = j - 1
            else
                matkm = p(i,j+1,2) - (p(i,j+1,2) - p(i,j,2)) *_
                        (p(i,j+1,1) - 1000) / (p(i,j+1,1) - p(i,j,1))
                ok$ = "yes"
            end if
        wend
    end if

'     compute the M-unit gradient for each profile

    for j = 1 to nmax(i) - 1
       delu = p(i,j+1,2) - p(i,j,2)
       delh = p(i,j+1,1) - p(i,j,1)
       if delh <> 0 then
           p(i,j,3) = 1.e-3 * delu / delh
       else
           p(i,j,3) = 1.e-6
       end if
    next j

'     compute the M-unit gradient at the top of the profile

    matsfc = p(i,1,2)
    h = p(i,nmax(i),1) / 1000
    a = log(matsfc / (matkm - 157))
    p(i,nmax(i),3) = 157 - (matsfc * a * exp(-a * h))
next i
end sub
```

END
DATE
FILMED
JAN
1988